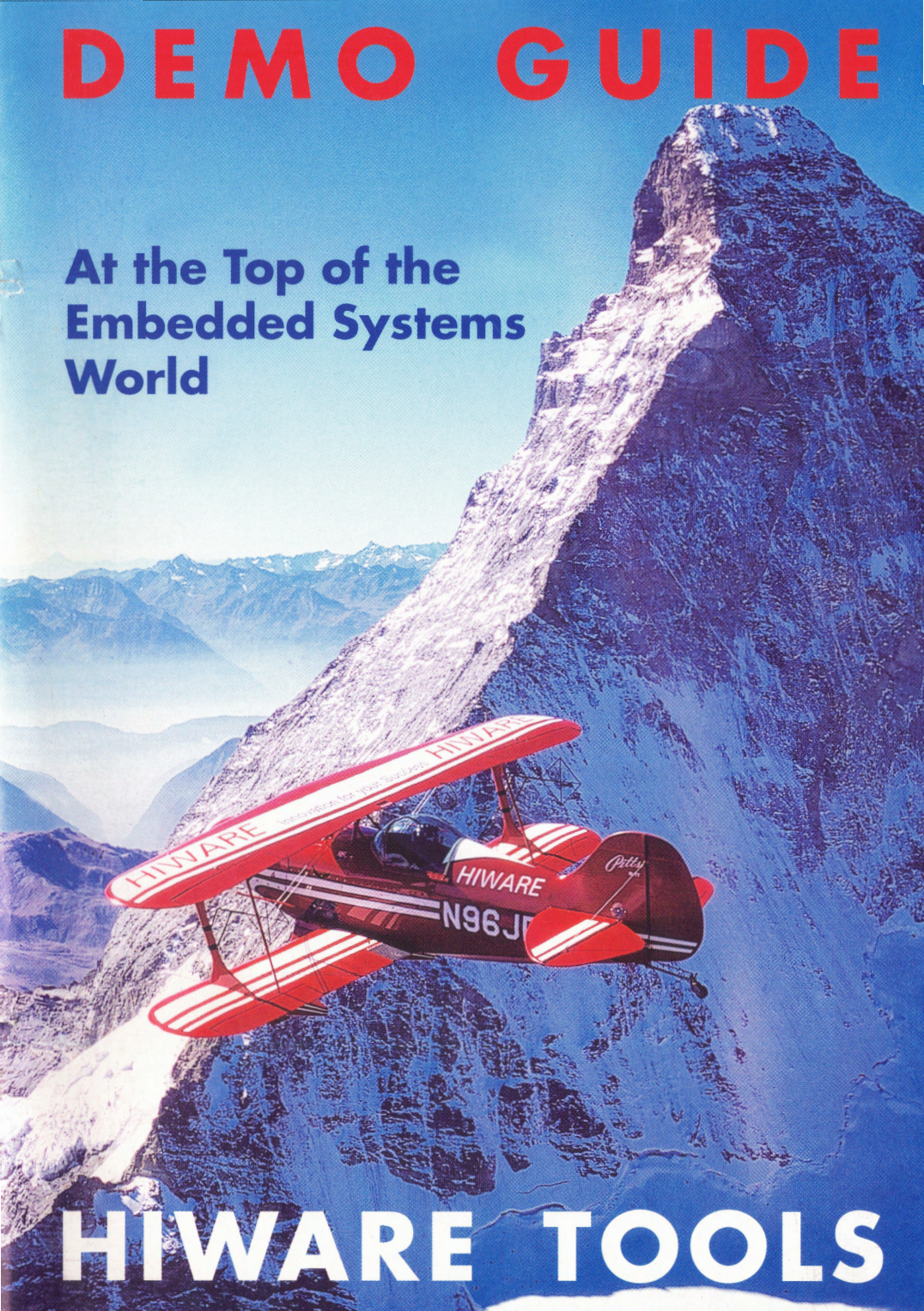
DEMO GUIDE

At the Top of the Embedded Systems World



HIWARE TOOLS

Quick Start

Introduction

This package contains:

- This *Quick Start* which will help you to quickly install this demo and to start to debug with *HI-WAVE*.
- The *Demo Guide M68HC11EVBU2*.
- *Manuals* delivered in *Adobe Acrobat Reader* format.
- *Release Notes* delivered in *Adobe Acrobat Reader* format.
- *HI-CROSS+ HC11 ANSI-C/cC++ Compiler*.
- The *HI-CROSS HC11-Assembler*.
- Optimizing smart *HI-CROSS Linker*.
- Object file/absolute file decoder/listing generator.
- The *HI-WAVE* development framework.
- The *WinEdit* source editor.

System Requirements

- A PC with an 80486 or better microprocessor is required. At least 16MB of RAM is recommended.
- A 3.5" disk drive and a hard disk with 12 MB storage available are required.
- Microsoft Windows (WIN32S, WIN95 or WINNT) is required.

Limitations of this demo kit

Compiler

- Maximum number of 10 functions for a source file.
- Maximum code size of 800 bytes for a source file.
- The banked memory model is not supported.
- A box is displayed whenever the compiler is started or a file is compiled reminding you that you are running a *limited demo version*.

Assembler

- Maximum number of 8 assembler sections.
- Code size limited to 1 KB.

Linker

The size of the generated `.abs` file is limited to 3 KB.

Debugger/Simulator

The maximum code size that can be loaded in the debugger is limited to 1 KB.

© *Copyright 1997 HIWARE*

Setting the M68HC11EVBU2 target board

Before connecting the *EVBU2* board to your system and running *HI-WAVE*, some jumpers must be set:
The *EVBU2* board has a built-in voltage regulator (5V), therefore it is possible to use an unregulated DC voltage (e.g. 12V). In this case set the jumper `J1 [2 3]`. You can also use an external regulated (5V) power supply and set the jumper `J1` to `[1 2]`.

Caution: An unregulated voltage with `J1` set to `[1 2]` can damage the board!

Jumpers `J3` and `J4` must be set to enable the monitor program downloading into the RAM, setting the *EVBU2* board in bootstrap mode.

Install

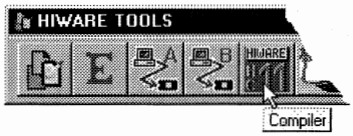
To install correctly the *Demo HIWARE TOOLS M68HC11EVBU2*, insert the disk labelled *Demo HIWARE TOOLS M68HC11EVBU2 disk1*. Use menu entry *Start* | *Run* of the windows task bar (for *Windows 95* or *NT 4.0*) or menu entry *File* | *Run* of the *File Manager* (for *Windows 3.11* or *NT 3.51*) to start `A:\Setup.exe`. The installation script will help you to install the demo software.

Compile - Link - Download - Debug

Follow these steps to compile, link and debug the `fibo.c` example. This will take you around 5 minutes:

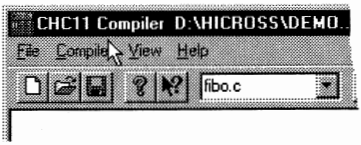
1. Connect the *M68HC11EVBU2* target board to the serial communication port of your PC and switch power on.
2. Launch the *HIWARE TOOLS* shell double-clicking on the icon:

3. Press the compiler button on the tool bar:



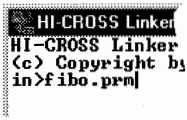
4. Compile the fibo program: type `fibo.c` in the edit box and press enter.

© *Copyright 1997 HIWARE*



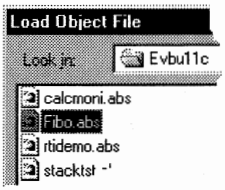
5. Close the compiler.
6. Start the linker: press the linker button on the tool bar.

7. Type in the name of the linker parameter file (`fibo.prm`).



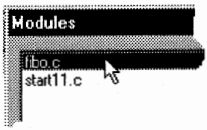
8. Press enter to link. The linker displays the message "`****** linking succeeded ******`" when the file has been successfully linked. You can then press ESC to close the linker.
9. Start the debugger pressing the *HI-WAVE* icon in the *HIWARE TOOLS* tool bar.

10. If the *M68HC11EVBU2* target board is connected to the serial port `COM1` of your `PC`, the *Download monitor* dialog is displayed. Press on the reset button on the target board. The monitor is then loaded. If the target board is connected to another serial port, an error box is displayed. When clicking *OK* it is closed and the *Communication Device Specification* dialog is opened. You can then specify the communication port used.
11. Select menu entry *Monitor* | *Load* to open the *Load Object File* dialog. Load the file `C:\HICROSS\DEMO\EVBU11C\fibo.abs`.

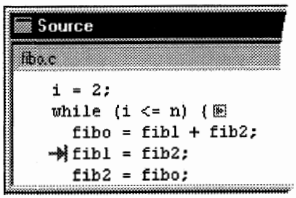


12. Point to the *Source* component and press the right mouse button. A pop up menu is displayed. In this menu, select *Open Module...*. The *Module* component is displayed.

© *Copyright 1997 HIWARE*



13. Double-Click on `fibo.c` to open the module fibo. The application source file is displayed in the *Source* component.
14. Point to a C source line in the *Source* component and press the right mouse button. In the Pop up menu select *Set Breakpoint*. A breakpoint is set on the corresponding C source line.



15. Run the application clicking →. The application runs until the breakpoint is encountered.
16. Close *HI-WAVE. Finished!*

Note: For further details on each of the above steps or how to automatize it (e.g. using a make file or working together with editors), please refer to the Demo Guide or the Manuals which are available in Adobe Acrobat Reader format in directory `C:\HICROSS\DOCU`.

© *Copyright 1997 HIWARE*

Demo Guide

M68HC11EVBU2

© Copyright 1997 HIWARE

Product Manual	Manual Date
Demo Guide M68HC11EVBU2	09/09/97 v1.0

© *Copyright 1997 HIWARE*

Contents

Demo Guide M68HC11EVBU2 5
Installation 5
Demo HIWARE TOOLS M68HC11EVBU2 Installation 5
Editor Installation 5
Win32s Installation 5
Overview 6
HIWARE TOOLS Demo 7
Project Directory 7
Starting with HIWARE TOOLS 7
Setting a New Project Directory 8
Editing a Project 10
Setting an Editor 10
Setting the Tools in HIWARE TOOLS Shell 14
Switching Between Project Directories 17
Assembler Demo 19
Switching to the Assembler Demo Project Directory 19
Starting the Assembler 19
Assembling a File 19
Assembler Options 20
Error Detection 21
Compiler Demo 22
Switching to the Compiler Demo Project Directory 22
Starting the C Compiler 22
Setting Compiler Options 23
Selecting the Input File 25
Getting Help on a Compiler Message 26
Create an Error Test File 26
Getting Help using the F1 Key 26
Getting Help using the Right Mouse Button 26
Error Feedback in an Editor 27
Error Feedback with WinEdit 27
Error Feedback with Notepad 28
Message Management 30
Disabling an Information Message 31
Enabling Back the Information Message 33
Linker Demo 34

© *Copyright 1997 HIWARE*

Linker Parameter File 34
Linking FIBO.C 35
Debugger/Simulator Demo 37
HI-WAVE 37
Starting the Debugger/Simulator 37
Starting the Debugger 37
Starting the Simulator 40
Switching Between Targets 40
Pull Down Menu in HI-WAVE 41
Loading and Starting an Application 41
Stopping an Application 43
Drag and Drop Facilities 44
To Drag and Drop an Object 44
Drag and Drop Combinations 44
View Where to Set Breakpoints 45
Working with Breakpoints 45
Setting Temporary Breakpoints 46
Setting Permanent Breakpoints 46
Stepping 47
Single Step on C Level 47
Step Over 47
Single Step on Assembler Level 47
Working with Data and Registers 47
Displaying and Updating Local Variables 47
Displaying and Updating Global Variables 48
Displaying and Updating Register Contents 50
Inspecting Memory 50
Working with the Assembler Window 51
On-line Disassembling 51
Consulting the Code 52
Working with Breakpoints in the Assembly Window 53
Simulating an Assembly Source File 53
Simulator Specific Features 54
Coverage Functions 54
Profiling Functions 55
Software Tracing Functions 56
Visualization Function 56
Debugging on the M68HC11EVBU2 Target Board 60
Using the *Terminal* Component 60
Debugging an Interrupt Driven Application 61
RTI Interrupt, C Based Application 61
RTI Interrupt, Assembler Application 62

© *Copyright 1997 HIWARE*

Demo Guide M68HC11EVBU2

Installation

Demo HIWARE TOOLS M68HC11EVBU2 Installation

To install correctly the *Demo HIWARE TOOLS M68HC11EVBU2*, insert the disk labelled *Demo HIWARE TOOLS M68HC11EVBU2 disk1*. Use menu entry *Start | Run* of the windows task bar (for *Windows 95* or *NT 4.0*) or menu entry *File | Run* of the *File Manager* (for *Windows 3.11* or *NT 3.51*) to start `A:\Setup.exe`. The installation script will help you to install the demo software.

By default, the installation is done in directory `C:\HICROSS`. The installation script allows you to browse for another directory. In this document the installation path will be `D:\HICROSS`.

Editor Installation

During the installation, if *WinEdit* is not present in your installation, the script prompts for the *WinEdit* installation disk. If *WinEdit* is installed, the installation script continues.

Win32s Installation

The development and debugging tools need *Win32s* to run under *windows 3.11*. If you are running this environment and if *Win32s* is not installed, the installation script will automatically prompt for the *Win32s* installation disks. *Win32s* will then be installed, the installation script will continue.

© *Copyright 1997 HIWARE*

Overview

This *Demo-Guide* will help you to use the *HIWARE* development and debugging tools with the *M68HC11EVBU2 target board.*

HIWARE TOOLS Demo

This demo shows you how to manage your project using the *HIWARE TOOLS* shell.

Assembler Demo

This demo shows you how to assemble an assembly source file and the *HI-CROSS HC11-Assembler* features.

Compiler Demo

This demo shows you how to compile a C source file and the *HI-CROSS+ HC11 Compiler* features.

Linker Demo

This demo shows you how to link an application.

Debugger/Simulator Demo

This demo gives an overview of the *HI-WAVE* debugger. Connecting the *M68HC11EVBU2* target board to the debugger is described here. This demo can be played using either the *M68HC11EVBU2* target board or the *Simulator.*

Simulator Specific Features

This demo gives an overview on some *Simulator* specific features. This demo can not be run on the *M68HC11EVBU2* target board.

Debugging on the M68HC11EVBU2 Target Board

This Demo gives examples that can only be run using the *M68HC11EVBU2* target board.

Note: For all demos, the installation directory is assumed to be `D:\HICROSS`.

© *Copyright 1997 HIWARE*

HIWARE TOOLS Demo

HIWARE TOOLS is a shell delivered with the *HI-CROSS+* package. *HIWARE TOOLS* helps you to configure your debugger environment and provides a shortcut bar that you can fully configure to develop and debug your applications.

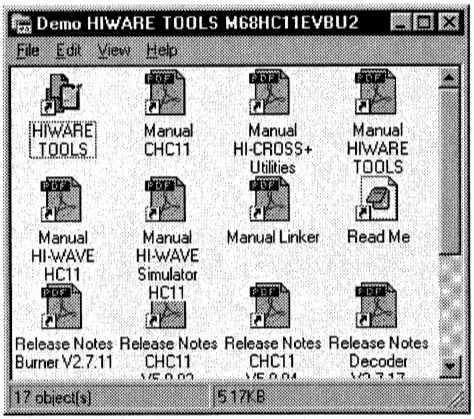
Project Directory

A project directory is a directory containing all environment files needed to configure your development environment.

Once you have performed the installation described previously, the project directory is automatically set to the *Demo ANSI-C M68HC11EVBU2* project directory. This directory contains initialization files that are required for the tools to work correctly. The path to this directory should be: `D:\HICROSS\DEMO\EVBU11C`.

Starting with HIWARE TOOLS

Double-click on the *HIWARE TOOLS* icon in the *Demo HIWARE TOOLS M68HC11EVBU2* group to run the *HIWARE TOOLS* shell or select *Start* | *Program* | *Demo HIWARE TOOLS M68HC11EVBU2* | *HIWARE TOOLS* from the Windows task bar.



The *HIWARE TOOLS* shell window pops up:

The *HIWARE TOOLS* shell window has properties identical to most of Windows applications. A tool tip is available for each of the buttons in the *HIWARE TOOLS* shell.

When clicking one of the *HIWARE TOOLS* shell button, the corresponding executable is started and is ready to run in the specified project directory.

In the *HIWARE TOOLS* shell, the button with *A* stands for *Assembler*, *B* for *Burner*, *D* for *Decoder*, *LIB* for *Librarian*, *L* for *Linker* and *M* for *Maker*.

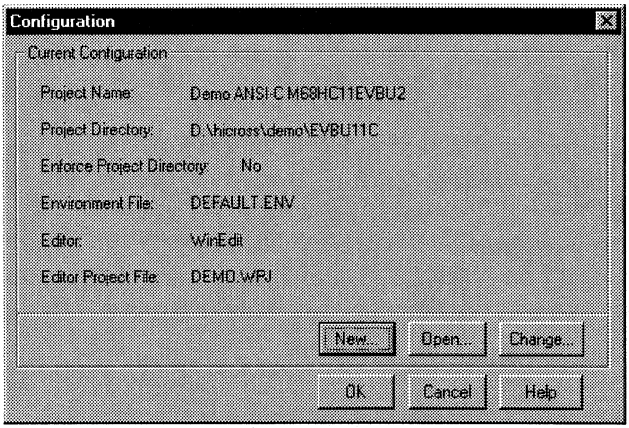
Click the *Assembler* button. The *HI-CROSS HC11 Assembler* is run and its main window is displayed. The Assembler environment is correctly set to assemble files from the project directory. Close the *Assembler* window.

Setting a New Project Directory

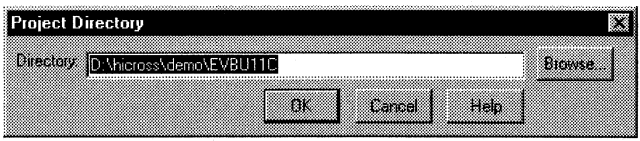
In order to write your own applications, you may need to set another project directory. This can be done using the *HIWARE TOOLS* shell. In that purpose, follow these steps:

1. When clicking the first icon of the bar, you open the *Configuration* dialog which enables to set your environment through different dialogs.

© *Copyright 1997 HIWARE*

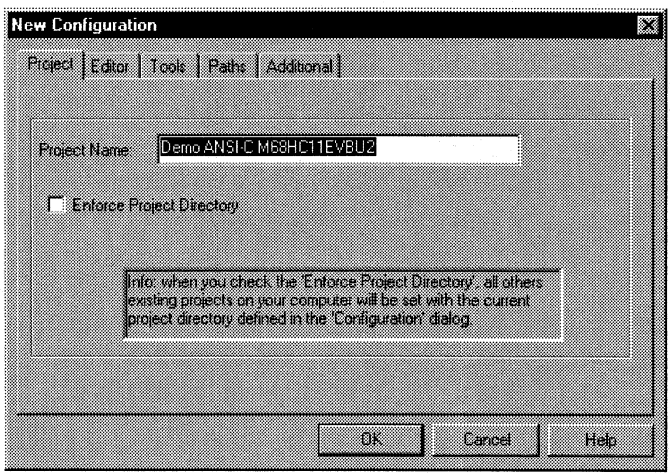


2. Click *New*. The *Project Directory* dialog pops up.



You can type in the edit box or browse for the directory path to set a new project directory. This new directory is created on your hard disk.

3. As an example type in `D:\HICROSS\DEMO\TEST`. When clicking *OK*, the *New Configuration* dialog is displayed. All the parameters of your project can be set, starting by the *Project Name* as shown below.



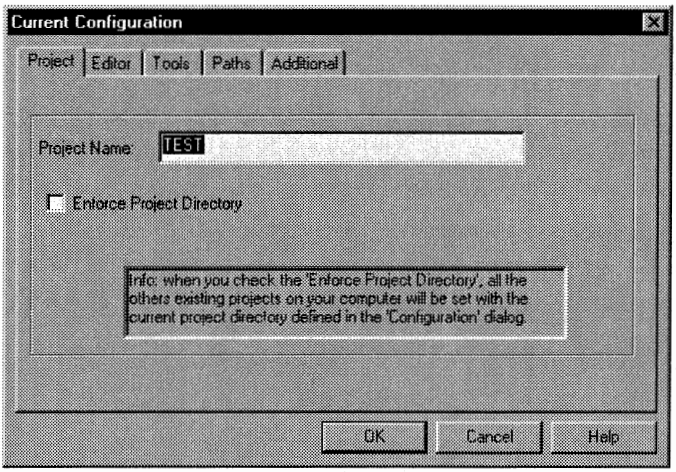
© *Copyright 1997 HIWARE*

4. Change the *Project Name* from *Demo ANSI-C M68HC11EVBU2* to `TEST`. Then you can progress from index to index and set all the parameters of your project. All the fields are initialized by default with the settings of the previous project directory.
5. When you click *OK*, the new project is recorded. In the *Configuration* dialog, the *Project Name* and *Project Directory* fields change: the *Project Name* is now `TEST` and the *Project Directory* is `D:\HICROSS\DEMO\TEST`.
6. Click *OK* in the *Configuration* dialog and *Yes* in the *ENVIRONMENT* dialog. The dialog are closed and the *HIWARE TOOLS* shell is set to run in the new project directory. All the needed initialization files have been stored in this new directory.

Editing a Project

Each project has its own properties and you can set all the tools and properties associated to your project with the *HIWARE TOOLS* shell.

From the *Configuration* dialog, click *Change...* to edit the current project and reach the following dialogs.



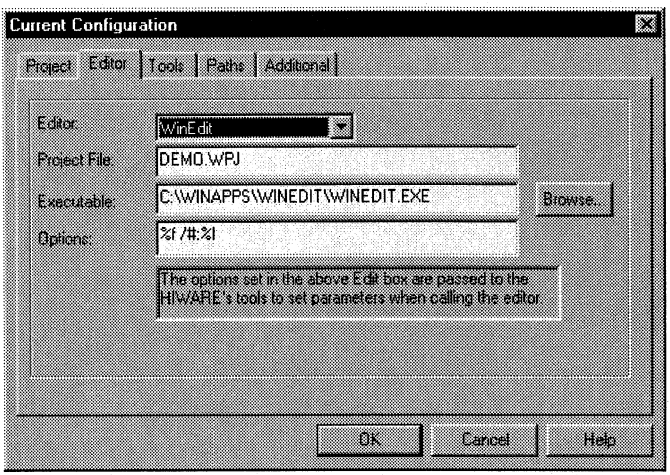
Setting an Editor

When installing the *Demo HIWARE TOOLS M68HC11EVBU2*, the *Editor* is set by default to *WinEdit*. If you want to use another editor, it must be set in the *HIWARE TOOLS* shell. As an example, we will replace *WinEdit* with *Notepad* in the

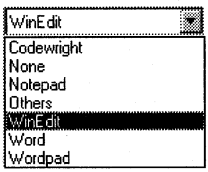
© *Copyright 1997 HIWARE*

HIWARE TOOLS shell.

1. Select the index *Editor* in the *Current Configuration* dialog. The settings for *WinEdit* are displayed in the index.

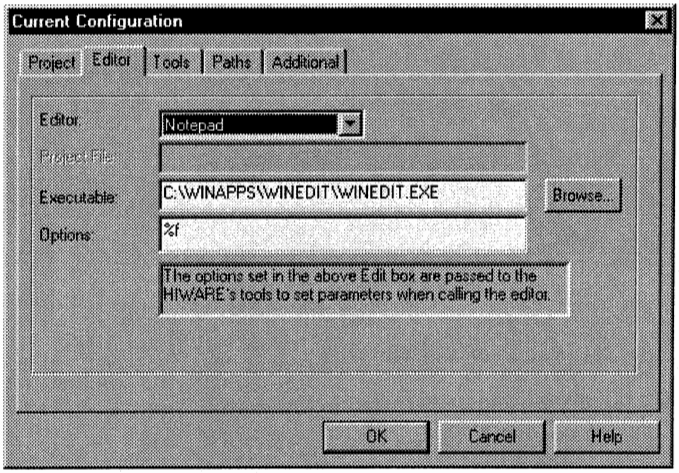


2. In the *Editor* combo box you can select one of the commonly used editor with the *HIWARE TOOLS* shell: *WinEdit*, *CodeWright* or a different Windows text editor.

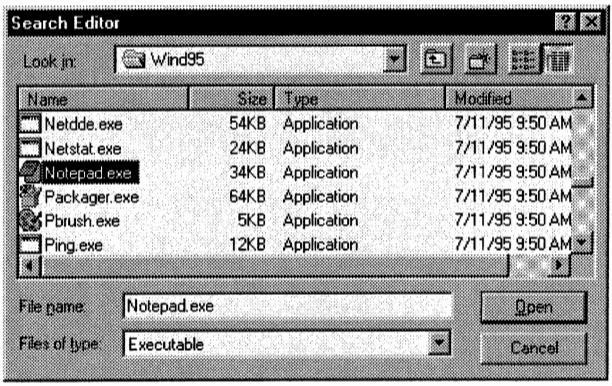


3. Select *Notepad* in the combo box.

© *Copyright 1997 HIWARE*

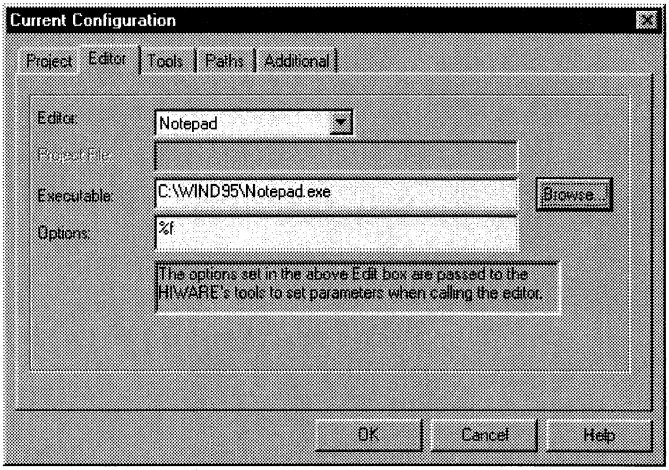


4. Now, you have to browse for the *Notepad* executable. Click *Browse*. The *Search Editor* dialog is displayed. Select the *Notepad* in your *Windows* installation directory and click *Open*.

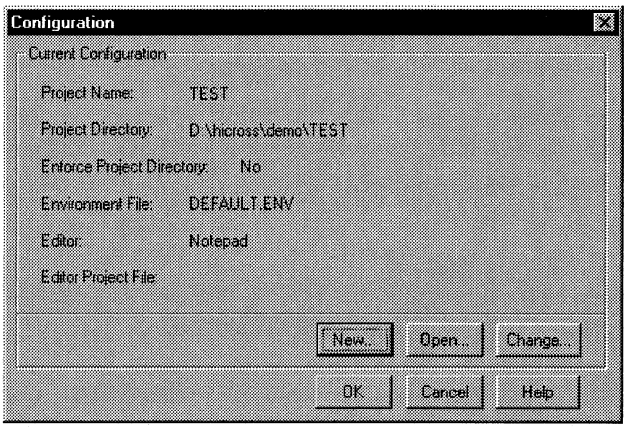


5. The *Current Configuration* dialog is updated with the new editor path.

© *Copyright 1997 HIWARE*



6. Click *OK* to close this dialog. The *Editor* field in the *Configuration* dialog is now set to *Notepad.*



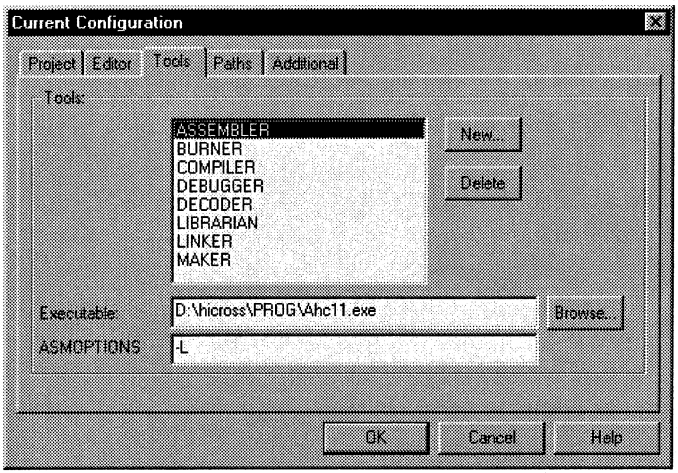
7. Click *OK* to close the *Configuration* dialog. Click *Yes* in the *ENVIRONMENT* dialog to update the configuration files.

8. In the *HIWARE TOOLS* shell, the *WinEdit* icon has been replaced with the *Notepad* icon. Click this icon to start the *Notepad.* Close the *Notepad.*

© *Copyright 1997 HIWARE*

Setting the Tools in HIWARE TOOLS Shell

Tools can be added or removed from the *HIWARE TOOLS* shell. In the *Configuration* dialog, click *Change...* to open the *Current Configuration* dialog. Select the index *Tools*.



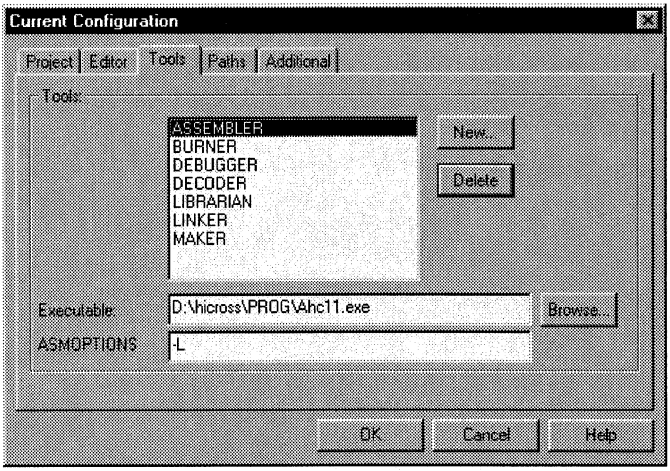
Applications visible in the shell tool bar are set in this dialog. When clicking on one of the name displayed in the list box, the corresponding application executable (including path) is displayed in the *Executable* edit box.

Removing a Tool

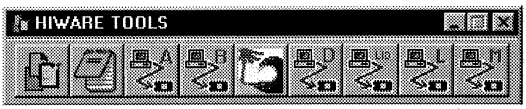
A tool can be easily removed from the *HIWARE TOOLS* shell. As an example, we will remove the *HI-CROSS+ CHC11 Compiler* from the tool bar.

1. Select `COMPILER` in the list box and click *Delete*. The `COMPILER` tool is removed from the *Tools* list box.

© *Copyright 1997 HIWARE*



2. Click *OK* to close the *Current Configuration* dialog.
3. Click *OK* to close the *Configuration* dialog. Click *Yes* in the *ENVIRONMENT* dialog to update the configuration files. The *HIWARE TOOLS* shell is now displayed, the *Compiler* icon has disappeared.

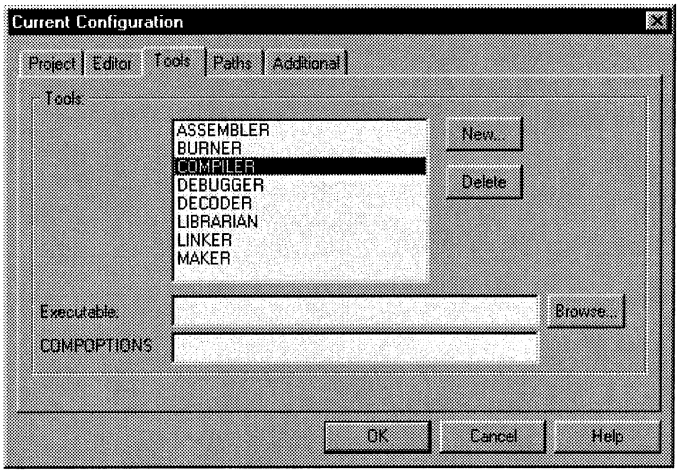


Adding a Tool

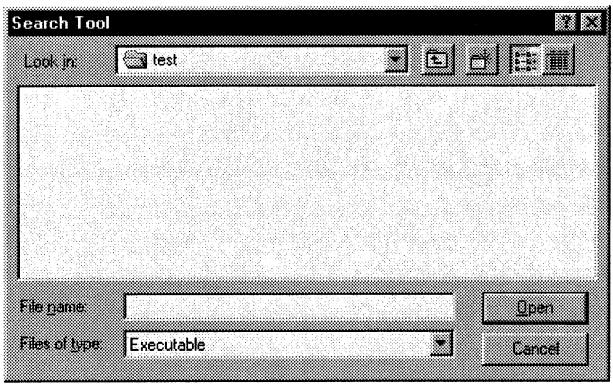
To set the Compiler back in the *HIWARE TOOLS* shell, follow these steps:

1. Open the *Configuration* dialog and click *Change...* to open the *Current Configuration* dialog. Select the index *Tools*.
2. Click *New...* the *New Tool* dialog is displayed.

3. Type in `COMPILER` and click *OK*. This name will be used for the tool tip in the short-cut bar.
4. `COMPILER` is added to the list of available tools in the list box.

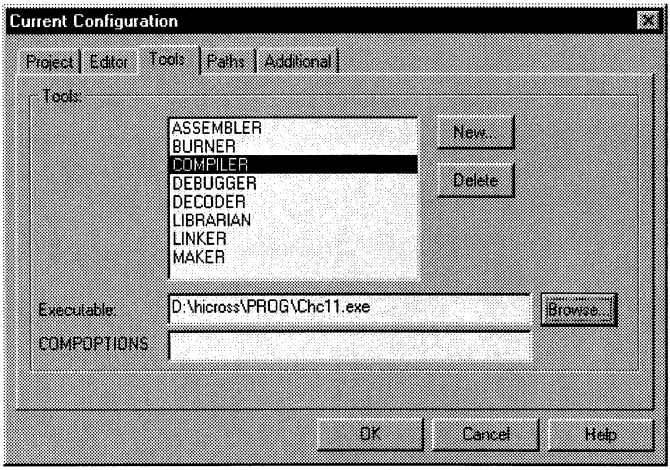


5. An executable has to be associated with the newly defined tools. This has to be done in the *Executable* edit box. The path for the executable can be directly entered or browsed. To associate one executable to the `COMPILER`, select `COMPILER` in the *Tools* list box and click *Browse...* . The *Search Tool* dialog is displayed.

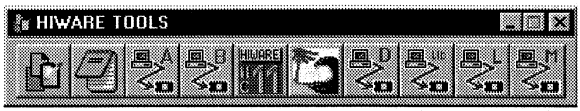


6. Use this dialog to browse for the *HI-CROSS+ CHC11 Compiler*. The path should be `D:\HICROSS\PROG\CHC11.EXE` . When the correct path has been set, click *Open* to close the *Search Tool* dialog. This path is displayed in the *Executable* edit box.

© *Copyright 1997 HIWARE*



7. Click *OK* to close the *Current Configuration* dialog.
8. Click *OK* to close the *Configuration* dialog. Click *Yes* in the *ENVIRONMENT* dialog to update the configuration files. The *HIWARE TOOLS* shell is displayed. The *HI-CROSS+ CHC11 Compiler* icon is back in the tool bar.



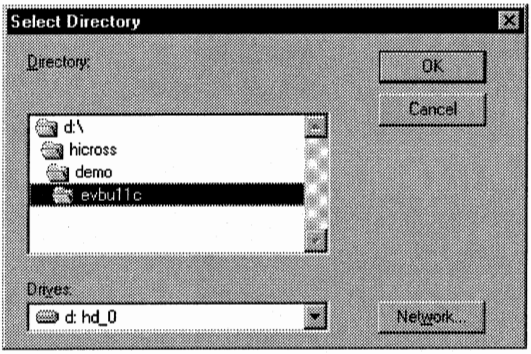
9. Click on this icon to start the Compiler. Choose menu entry *File* | *Exit* to quit the Compiler.

If the *HIWARE TOOLS* shell cannot find an application defined in the *Executable* path, the icon shown below is set in the short-cut bar.

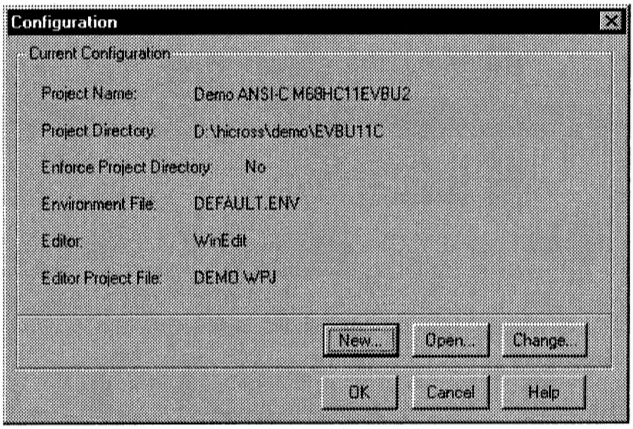
Switching Between Project Directories

To switch back from the `TEST` project directory to the `Demo ANSI-C M68HC11EVBU2` project directory (to open an already defined project directory), follow these steps:

1. In the *Configuration* dialog, click *Open...* The *Select Directory* dialog is opened. Browse for the directory containing the `Demo ANSI-C M68HC11EVBU2` demo. The path should be `D:\HICROSS\DEMO\EVBU11C`.



2. Click *OK*. The *Select Directory* dialog is closed.



3. In the *Configuration* dialog, the *Project Name*, *Project Directory* and *Editor* fields change: the *Project Name* is now `Demo ANSI-C M68HC11EVBU2`, the *Project Directory* `D:\HICROSS\DEMO\EVBU11C` and the *Editor* is *WinEdit*. Click *OK* to close the *Configuration* dialog. The *HIWARE TOOLS* shell is ready to work in the *Demo ANSI-C M68HC11EVBU2* project directory.

© *Copyright 1997 HIWARE*

Assembler Demo

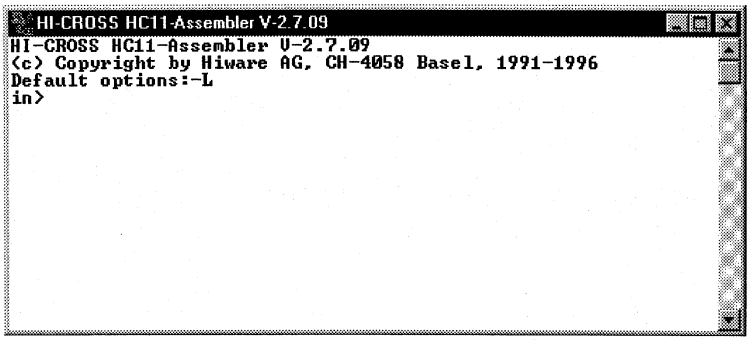
Switching to the Assembler Demo Project Directory

To perform the Assembler demo, use the *HIWARE TOOLS* shell to switch to the `Demo Assembler M68HC11EVBU2` project directory. The path to this directory is: `D:\HICROSS\DEMO\EVBU11A`.

Starting the Assembler

Start the *HIWARE TOOLS* shell and click the button corresponding to the *Assembler* to start the *HI-CROSS Assembler*.

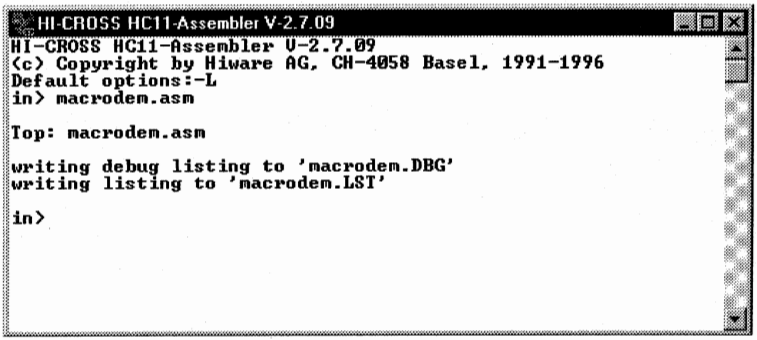
The main *HI-CROSS HC11-Assembler* window is displayed.



Assembling a File

To assemble the source file, type in its name and press enter. The file is immediately assembled. As an example, type in `macrodem.asm`

© *Copyright 1997 HIWARE*

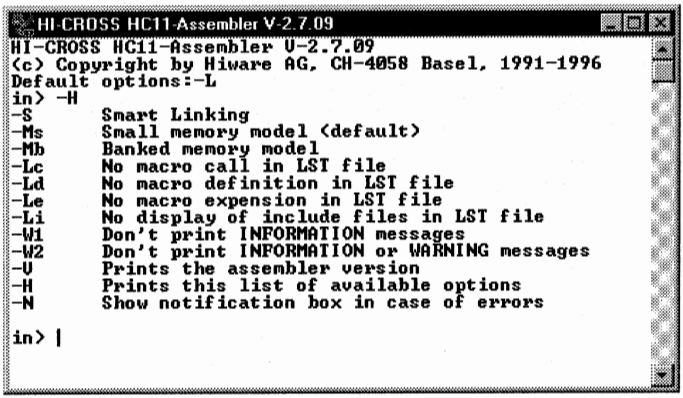


The file `macrodem.asm` has been successfully assembled. A debug listing file (`.dbg`), an object file (`.o`) and a listing file (`.lst`) have been generated and saved in the project directory.

Assembler Options

The listing file is optional: to generate it, the option `-L` must be set. The Macro Assembler options can be set or reset using the command line or within the shell.

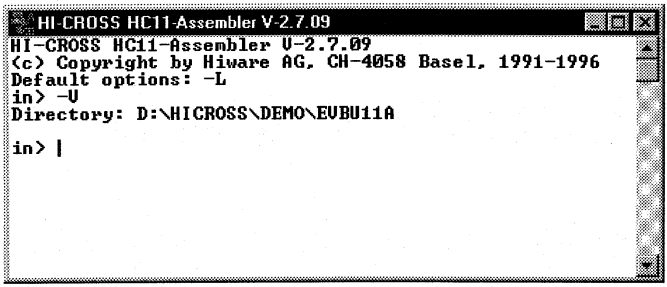
To obtain the list of the different options existing for the *HI-CROSS HC11-Assembler*, type `-H` in the *Assembler* main window.


HI-CROSS HC11-Assembler V-2.7.09
HI-CROSS HC11-Assembler V-2.7.09
(c) Copyright by Hiware AG, CH-4058 Basel, 1991-1996
Default options:-L
in> -H
-S Smart Linking
-Ms Small memory model (default)
-Mb Banked memory model
-Lc No macro call in LST file
-Ld No macro definition in LST file
-Le No macro expension in LST file
-Li No display of include files in LST file
-W1 Don't print INFORMATION messages
-W2 Don't print INFORMATION or WARNING messages
-V Prints the assembler version
-H Prints this list of available options
-N Show notification box in case of errors
in>

The list of the available options is displayed.

To find out the actual project directory, the option `-V` can be use.

© *Copyright 1997 HIWARE*



When typing `-V` in the *Assembler* window, the current project directory path is displayed.

Error Detection

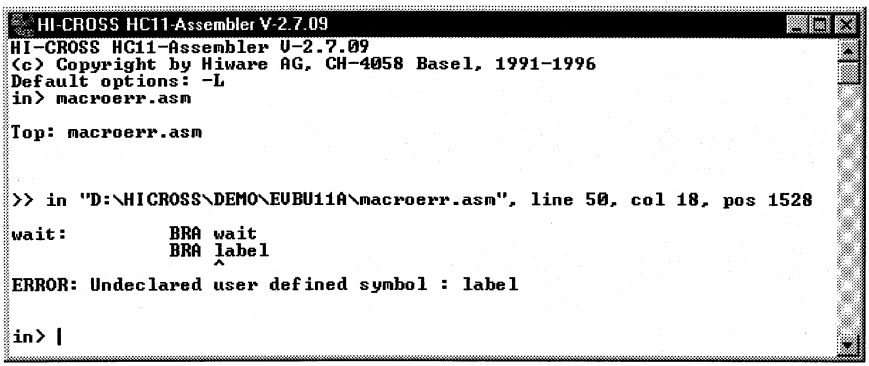
Using *WinEdit*, open `MACRODEM.ASM` in the `D:\HICROSS\DEMO\EVBU11A` directory.

Add somewhere in the code:

```
BRA label
```

Save the file as `MACROERR.ASM` and close the editor.

Assemble this file using the *HI-CROSS HC11-Assembler*



The Macro Assembler has detected an error while assembling the file. Information are given on the error type and location.

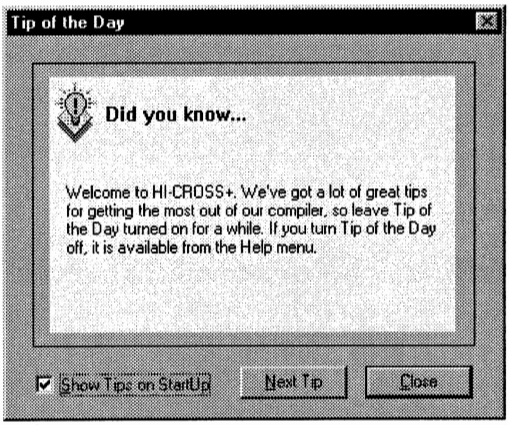
Compiler Demo

Switching to the Compiler Demo Project Directory

To perform the Compiler demo, use the *HIWARE TOOLS* shell to switch to the `Demo ANSI-C M68HC11EVBU2` project directory. The path to this directory is: `D:\HICROSS\DEMO\EVBU11C.`

Starting the C Compiler

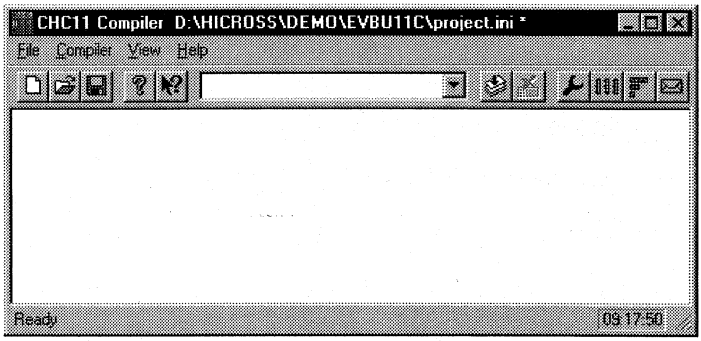
Start the *HIWARE TOOLS* shell and click the *Compiler* button to start the C Compiler. The *Tip of the Day* dialog box is opened.


Tip of the Day
Did you know...
Welcome to HI-CROSS+. We've got a lot of great tips for getting the most out of our compiler, so leave Tip of the Day turned on for a while. If you turn Tip of the Day off, it is available from the Help menu.
Show Tips on StartUp
Next Tip
Close

Click *Close* to close this dialog box.

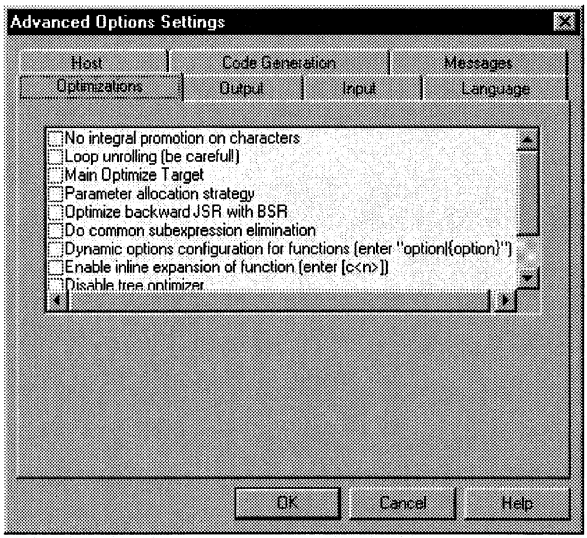
The main *HI-CROSS+ Compiler* window is displayed.

© *Copyright 1997 HIWARE*



Setting Compiler Options

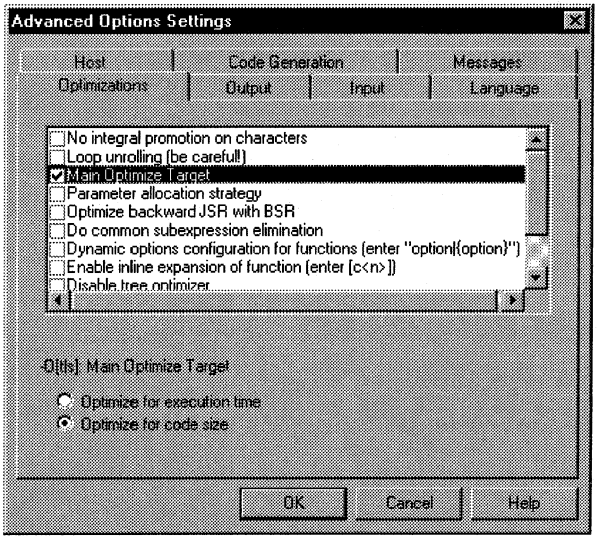
Compiler options can be set or reset choosing *Compiler* | *Options* | *Advanced* or clicking . The *Advanced Options Settings* dialog is opened.



Different indexes are available for Compiler option settings. To set a Compiler option, check the box in front of the desired option. More information could be required and set in the lower part of the index page. When the correct settings have been specified, close the dialog using *OK* to set the changes.

For further option details, please refer to the *Compiler* manual.

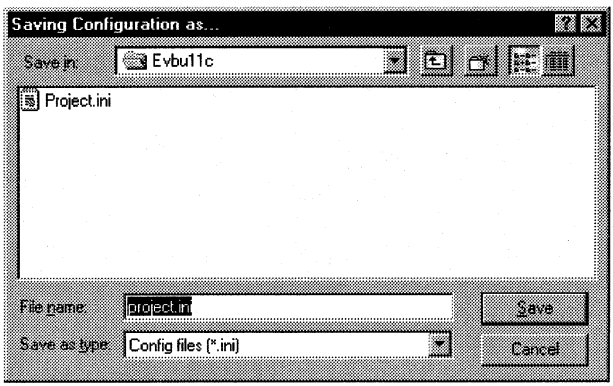
© Copyright 1997 HIWARE



To obtain *Help* on an option, select this option an press the F1 key. The corresponding topic of the *HelpFile* is automatically opened.

Those settings may be saved using the menu entry *File* | *Save Configuration as...* .

The *Saving Configuration as...* dialog is entered.



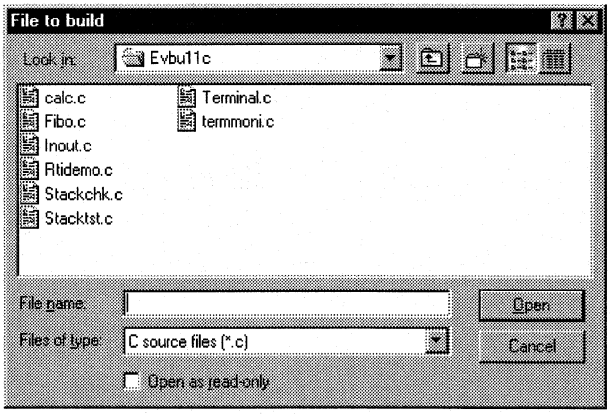
You can specify a file name for your configuration file in the edit box (`MyConfig.INI`) and click *Save*.

By default, you can save these options in your current project initialization file: `PROJECT.INI`.

Selecting the Input File

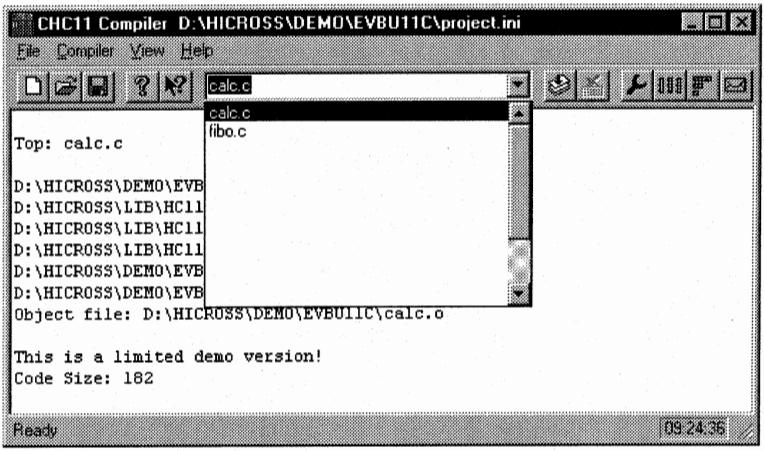
To select the input file which must be compiled you have the following possibilities:

- Select *File* | *Compile*. The *File to build* dialog box is opened. Select the file you want to compile, `D:\HICROSS\DEMO\EVBU11C\CALC.C` and click *Open*. The file is automatically compiled.



- Start the *Explorer* or the *File manager* and switch to the directory `D:\HICROSS\DEMO\EVBU11C`. Select the file `CALC.C` and drag it to the *Compiler* window. The file is compiled as soon as you drop it.
- Another way to compile a file is to use the edit box and the associated list from the *Compiler* window tool bar. Type in directly the name of the file to compile in the edit box or select it from the listbox and click in the tool bar or press ENTER to compile the file.

© *Copyright 1997 HIWARE*



Getting Help on a Compiler Message

Create an Error Test File

Using *WinEdit*, open `FIBO.C` in the `D:\HICROSS\DEMO\EVBU11C`.

In the main of the program, put in comment variable i declaration: `// int i;`

Save the file as `FIBOERR.C` and close the editor.

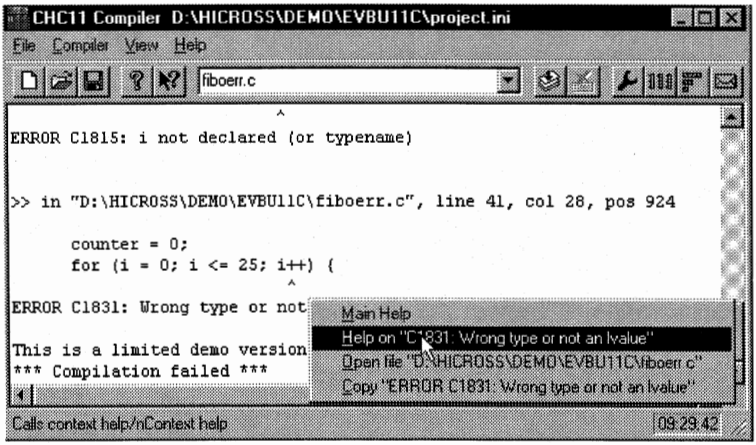
Getting Help using the F1 Key

- Once the file `FIBOERR.C` has been compiled, errors are generated.
- Click one of the lines where the source line generating the error message is displayed and press F1. The help file is opened and shows a description of the error message.
- Press ESC to close the help file.

Getting Help using the Right Mouse Button

- Once the file `FIBOERR.C` has been compiled, errors are generated.
- Points to the code line generating the error message and click the right mouse button. A *Context Menu* pops up.

© *Copyright 1997 HIWARE*


CHC11 Compiler D:\HICROSS\DEMO\EVBU11C\project.ini
File Compiler View Help
fiboerr.c
ERROR C1815: i not declared (or typename)
>> in "D:\HICROSS\DEMO\EVBU11C\fiboerr.c", line 41, col 28, pos 924
counter = 0;
for (i = 0; i <= 25; i++) {
ERROR C1831: Wrong type or not
This is a limited demo version
*** Compilation failed ***
Main Help
Help on "C1831: Wrong type or not an lvalue"
Open file "D:\HICROSS\DEMO\EVBU11C\fiboerr.c"
Copy "ERROR C1831: Wrong type or not an lvalue"
Calls context help/nContext help
09:29:42

- This menu contains four entries.
  - When selecting the *Main Help* entry, the Main help file topic is loaded.
  - When selecting *Help on ...* the help file is opened and shows a description of the error message.
  - When selecting *Open File ...* the source file is loaded in the specified editor (at the moment *WinEdit*). The `fiboerr.c` file is opened in *WinEdit* and the cursor reaches automatically the line producing the error.
  - The last entry of the pops up menu *Copy ...* is to be used for *Error Feedback* in editor that does not allow a line number as parameter.

Error Feedback in an Editor

Error Feedback with WinEdit

When compiling the file `FIBOERR.C`, errors are issued. If *WinEdit* is set as *Editor*, Error Feedback can be done using two ways:

1. Double-click one of the line where the source line generating the error message is displayed.:
   - *WinEdit* is started and the source file is automatically opened.
   - The cursor reaches the line where the error message has been issued.
   - Close *WinEdit*.
2. Using the *Context Help* pop up menu:

© *Copyright 1997 HIWARE*

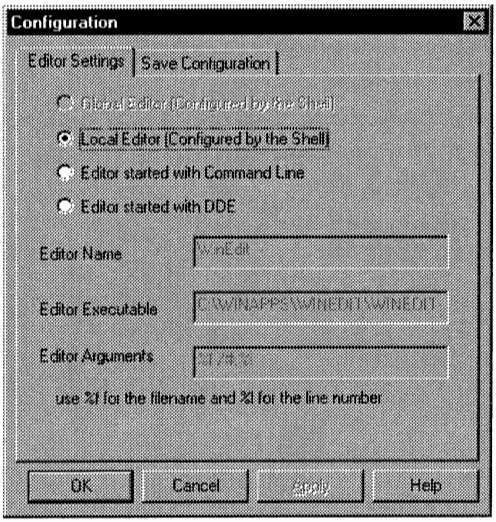
- Points to the line generating the error and click the right button.
- The *Context Help* pop up menu is displayed. Select *Open File ...*
- The source file is loaded in *WinEdit*. The cursor reaches the line where the error message has been issued.
- Close *WinEdit*.

Error Feedback with Notepad

Configure the C Compiler to Use Notepad as Editor

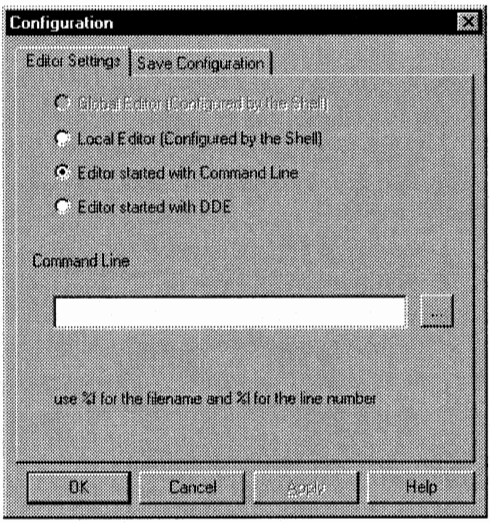
WinEdit has been set in the *HIWARE TOOLS* shell as the editor use by default for all the tools. The Editor settings can be forced locally to use another editor for the HC11 Compiler. As an example, we will set the *Notepad* as an *Editor Started with Command Line*.

1. Choose menu entry *File | Configuration ....* The *Configuration* dialog is opened. The default *Editor Settings* are displayed. Currently, the C Compiler is set to use the editor configured in the *HIWARE TOOLS* shell.

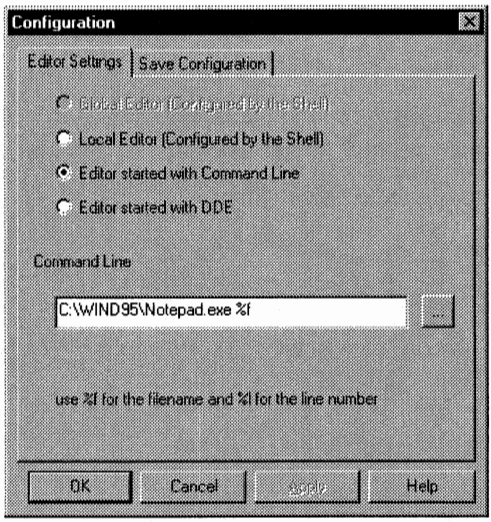


2. Select the *Editor Started with Command Line* radio button.

© *Copyright 1997 HIWARE*



3. In the *Command Line* edit box enter the path for *Notepad* (in your Windows directory) or browse for it using ... . In the command line, add the `%f` modifier to the path for the *Notepad* executable as shown below so that the source file will be automatically loaded in the editor.



4. Click *OK* to close the *Configuration* dialog.

© Copyright 1997 HIWARE

Error Feedback using Notepad

1. Compile the file `FIBOERR.C`. Errors are issued.
2. Click in the *Compiler* window one of the source line generating an error message.
3. Press CTRL + C or use the *Context Help* pop up menu entry *Copy ...* to copy the current line in the clipboard.
4. Double-click this line or use the *Context Help* pop up menu entry *Open ...* the *NotePad* is started and the source file is automatically opened.
5. Select *Search* | *Find*, the *Find* dialog box is opened.
6. Press CTRL + V to copy the content of the clipboard in the edit box, then click *Find*. *NotePad* will jump to the line, where the error message has been issued.
7. Close *NotePad*.

Switching Back to the WinEdit Editor

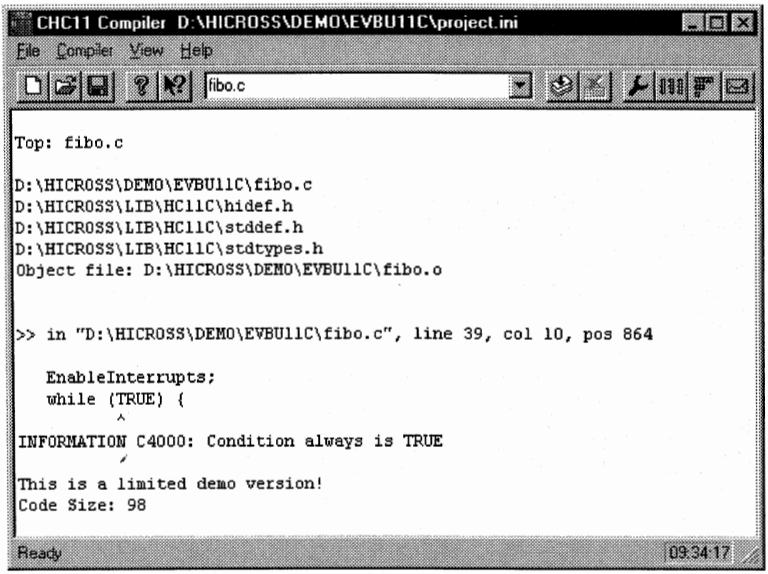
1. Choose menu entry *File* | *Configuration ....*
2. In the *Configuration* dialog, select the *Local Editor (Configured by the shell)* radio button. Click *OK*.
3. The *Configuration* dialog is closed, the Compiler is now using the *WinEdit* as editor.

Message Management

When compiling a file, the Compiler may issue a message. There are four classes of messages *Fatal*, *Error*, *Warning*, *Information*. A message can also be *Disabled*.

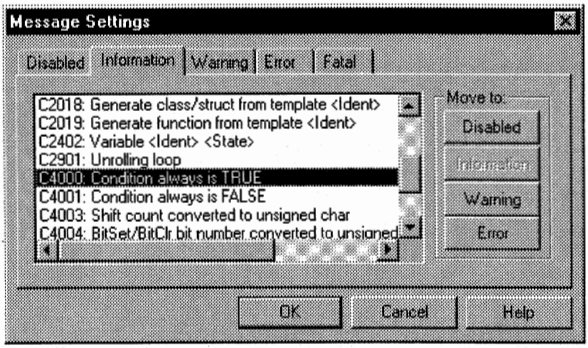
When compiling `FIBO.C`, an *Information* Message is issued:

© *Copyright 1997 HIWARE*

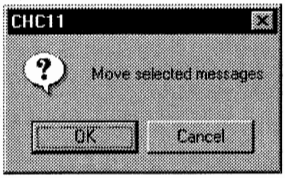


Disabling an Information Message

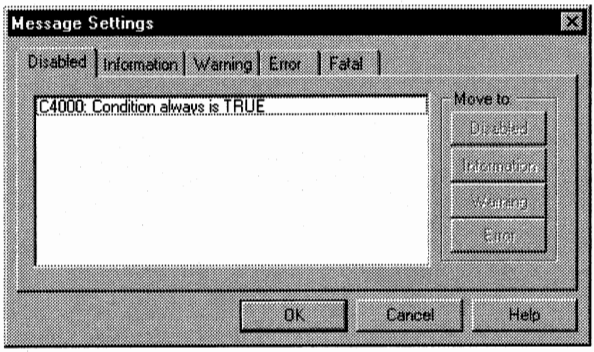
- Click or choose *Compiler* | *Messages*. The *Message Settings* dialog is entered.
- This dialog allows to switch the message class. Select `C4000: Condition always is TRUE` message in the *Information* index. This is the message generated when compiling `FIBO.C`.



- Click the *Disabled* button. This dialog pops up:

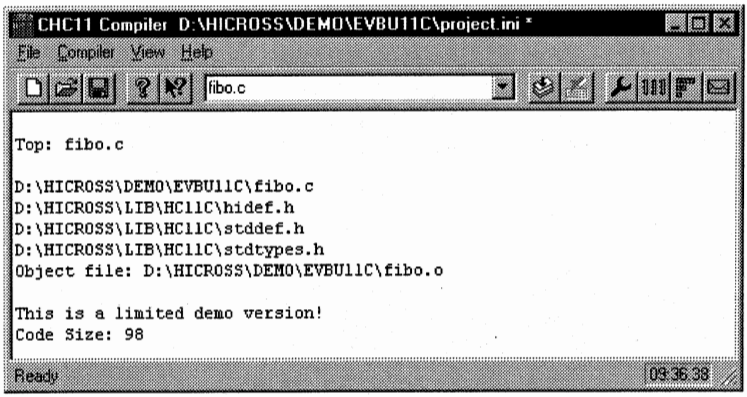


- Click *OK*. The `C4000` message disappears from the list of *Information* messages.
- Select the *Disabled* folder. The `C4000` message is displayed here. This Folder contains all the disabled messages.



- Click *OK* in the *Message Settings* dialog.

When compiling `FIBO.C`, no information message is generated anymore.



© *Copyright 1997 HIWARE*

Enabling Back the Information Message

To enable the `C4000` information message back:

- Open the *Message Settings* dialog and select the `C4000` message in the *Disabled* folder.
- Click the *Information* button from the *Move to:* button group. The Move selected messages dialog pops up.
- Click *OK*. The `C4000` information message is removed from the *Disabled* folder.
- Select the *Information* Folder. The `C4000` message can be retrieved in the information messages list.
- To enable this message back, click *OK* in the *Message Settings* dialog.
- Choose *File* | *Exit* to close the *HI-CROSS+ Compiler*.

© *Copyright 1997 HIWARE*

Linker Demo

Once a file has been assembled or compiled it needs to be linked so that the simulator or debugger can run it. This demo should be performed in the `D:\HICROSS\DEMO\EVBU11C` project directory.

Linker Parameter File

The input file for the Linker is a parameter file. The extension used is `.PRM`.

As an example `FIBO.PRM`, the parameter file corresponding to the `FIBO.C` file we compiled previously, looks like:

```
LINK fibo.abs
NAMES  fibo.o start11.o ansi.lib END
SECTIONS
  MY_ZPAGE = READ_WRITE 0x0080 TO 0x008F;
  MY_RAM   = READ_WRITE 0x0090 TO 0x018F;
  MY_ROM   = READ_ONLY  0xF000 TO 0xFDFF;
  MY_STK   = READ_WRITE 0x0190 TO 0x01FF;
PLACEMENT
  DEFAULT_ROM                  INTO  MY_ROM;
  DEFAULT_RAM                  INTO  MY_RAM;
  zpage                        INTO  MY_ZPAGE;
  SSTACK                       INTO  MY_STK;
END
VECTOR 0 _Startup /* set reset vector on startup function */
```

`LINK fibo.abs` Name of the generated executable.

The executable file produced by the Linker will be called `FIBO.ABS`.

`NAMES ... END` Names of the object files in the application.

The file `FIBO.ABS` uses procedures or global variables from the files `FIBO.O`, `START11.O` and `ANSI.LIB`.

`SECTIONS` Memory Map: defines RAM and ROM areas.

Sections `MY_ZPAGE`, `MY_RAM`, `MYROM` and `MY_STK` are defined.

`PLACEMENT ... END` Assignment of the different sections in the memory.

The code of the file `FIBO.ABS`, the strings and constant variables will be placed into section `MY_ROM`. The global data of `FIBO.ABS` will be placed into section

© *Copyright 1997 HIWARE*

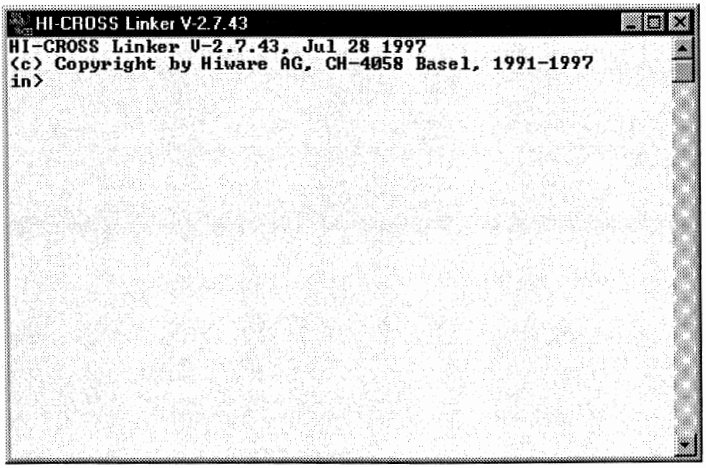
MY_RAM. The variable that must be placed on page zero are placed in zpage. The stack will be placed in SSTACK.

VECTOR Initialise reset vector.

For more information on the parameter files, please refer to *Linker* documentation.

Linking FIBO.C

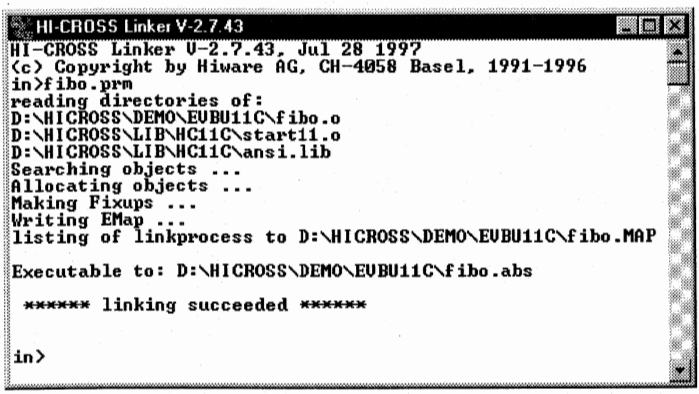
- To start the Linker, click the corresponding button in the *HIWARE TOOLS* shell. The *Linker* window is opened and you are prompted for a Linker Parameter File name.



- On the command line type FIBO.PRM.

© *Copyright 1997 HIWARE*

- The Linker displays the following information in the main window:



- Linking is successful. Press ESC to close the *HI-CROSS Linker* window.

© *Copyright 1997 HIWARE*

Debugger/Simulator Demo

The actions described in this chapter should be performed either using the debugger (*Smonitor* target) or the simulator (*Sim* target).

HI-WAVE

HI-WAVE is a Multipurpose Tool that can be used for various tasks in the embedded system and industrial control world. Some typical tasks are:

- Simulation and debugging of an embedded application.
- Simulation and/or cross-debugging of an embedded application.
- Simulation of a hardware design (e.g., board, processor, I/O chip).
- Building a target application using an object oriented approach.
- Building a host application controlling a plant using an object oriented approach.

Starting the Debugger/Simulator

Depending on the project directory selected, the debugger or the simulator is started.

Starting the Debugger

To run this demo using the *Debugger*, switch to the project directory `D:\HICROSS\DEMO\EVBU11C` using the *HIWARE TOOLS* shell.

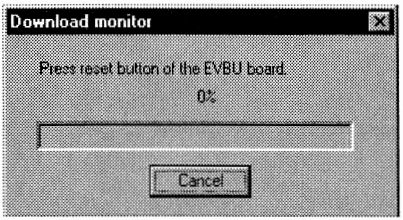
Click the *Debugger* button:

in the *HIWARE TOOLS* shell.

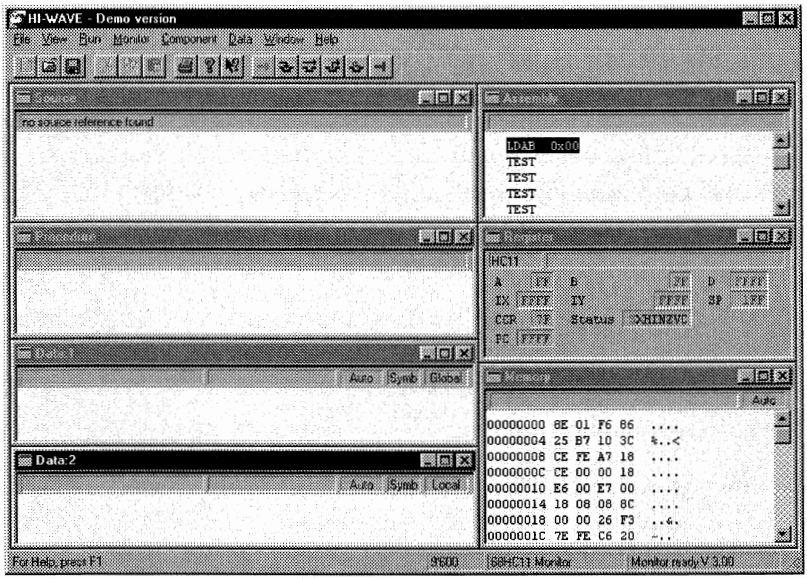
M68HC11EVBU2 Board Connected to a Serial Port

If the *M68HC11EVBU2* is connected and if the communication is established between the target board and the PC, the *Download monitor* dialog is displayed:

© *Copyright 1997 HIWARE*

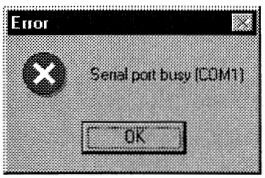


press on the reset button on the target board. The *Monitor* target is loaded.



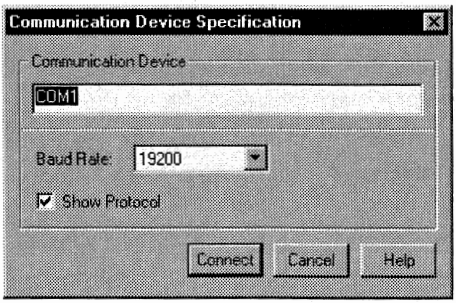
Monitor ready V3.00 indicates in the status line that the target is correctly set, an application can be loaded in the debugger.

When starting the Debugger, this *Error* dialog box should be first displayed in place of the *Download monitor* dialog:



Click *OK* in this dialog. The *Communication Device Specification* dialog is opened.

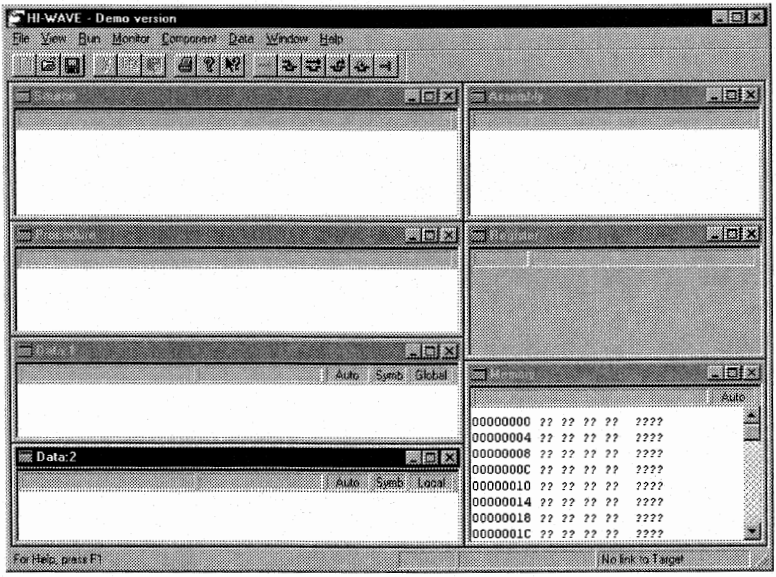
© *Copyright 1997 HIWARE*



In the *Communication Device* edit box, specifies the communication port used for the *M68HC11EVBU2* and click *Connect.* The *Download monitor* dialog is displayed. When pressing on the reset button on the *M68HC11EVBU2* target board, the *Monitor* target is loaded.

Target Board Not Connected

If the *M68HC11EVBU2* is not connected to your *PC*, click *Cancel* in the *Download monitor* dialog or *OK* in the error dialog box. The *Communication Device Specification* dialog is opened. Click *Cancel* in this dialog. *HI-WAVE* is then opened.


HI-WAVE - Demo version
File View Run Monitor Component Data Window Help
Data:2
Auto Symb Local
00000000 ?? ?? ?? ?? ????
00000004 ?? ?? ?? ?? ????
00000008 ?? ?? ?? ?? ????
0000000C ?? ?? ?? ?? ????
00000010 ?? ?? ?? ?? ????
00000014 ?? ?? ?? ?? ????
00000018 ?? ?? ?? ?? ????
0000001C ?? ?? ?? ?? ????
For Help, press F1
No link to Target

No link to Target indicates in the status line that the target has to be set.

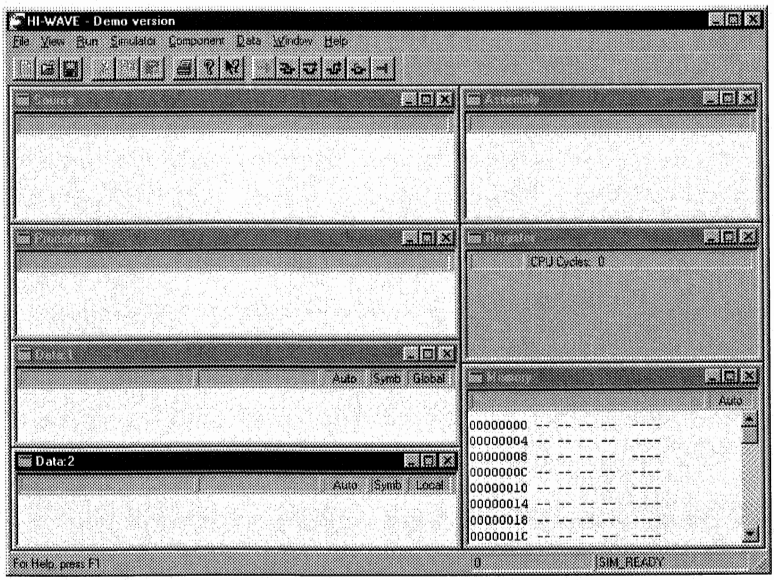
Starting the Simulator

To run this demo using the *Simulator*, switch to the project directory `D:\HICROSS\DEMO\SIM11C` using the *HIWARE TOOLS* shell.

Click the *Debugger* button:

in the *HIWARE TOOLS* shell.

The application is started and the simulator target is set.


HI-WAVE - Demo version
File View Run Simulator Component Data Window Help
CPU Cycles: 0
Auto Symb Global
Data:2
Auto Symb Local
Auto
00000000 -- -- -- -- ----
00000004 -- -- -- -- ----
00000008 -- -- -- -- ----
0000000C -- -- -- -- ----
00000010 -- -- -- -- ----
00000014 -- -- -- -- ----
00000018 -- -- -- -- ----
0000001C -- -- -- -- ----
For Help, press F1
0
SIM_READY

SIM_READY indicates in the status line that the simulator is ready.

Switching Between Targets

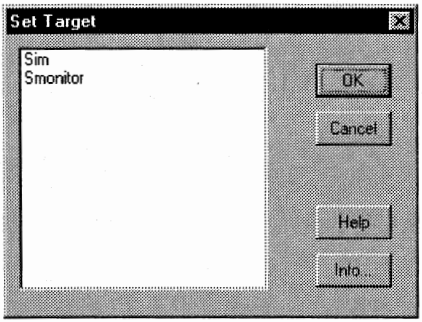
The *M68HC11EVBU2* (*Smonitor*) target and the *Simulator* targets are currently delivered with this demo version.

- *Smonitor* must be set when you want to debug on the *M68HC11EVBU2* target board.
- *Sim* must be set when you want to debug using the *Simulator*.

© *Copyright 1997 HIWARE*

When running the *Smonitor* target, to switch to the *Simulator* target, use the following steps:

1. Select menu entry *Component* | *Set Target*. The *Set Target* dialog is opened.



2. Select the desired target: to use the simulator, select *Sim*, to use the *Evaluation Board*, select the *Smonitor*. At the moment, we will use the simulator. So select *Sim* and click *OK*.
3. *SIM_READY* is displayed in the status bar. The target is now the simulator.

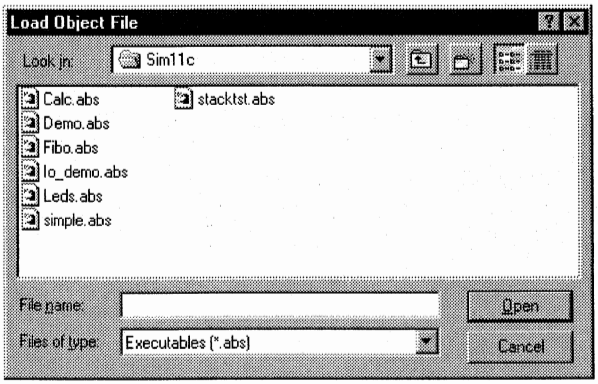
Pull Down Menu in HI-WAVE

Clicking the right mouse button in a component opens the corresponding pop-up menu, containing the functions associated with the component.

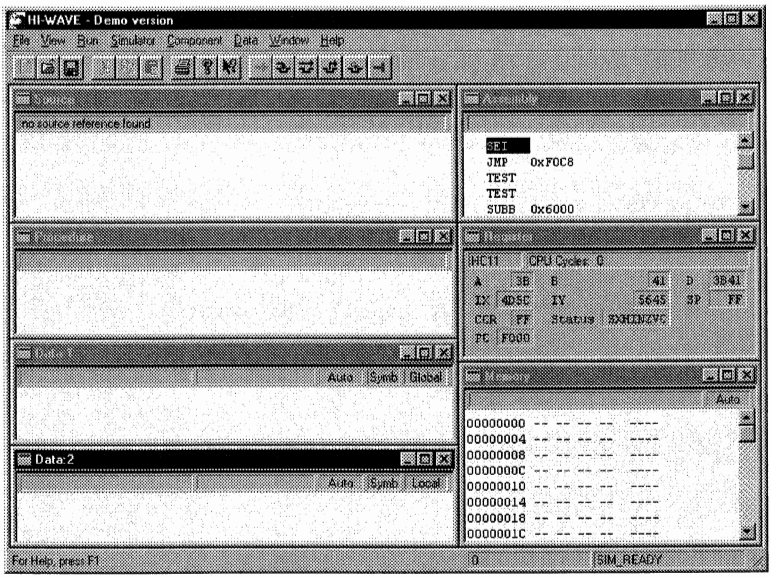
Loading and Starting an Application

1. Choose *Simulator* | *Load* or *Monitor* | *Load*. The *Load Object File* dialog is then opened.

© *Copyright 1997 HIWARE*



2. Select the file *FIBO.ABS* and click *Open.*
3. The dialog is closed and the program is loaded.



4. Choose *Run|Start/Continue* or click .
5. The application is started.

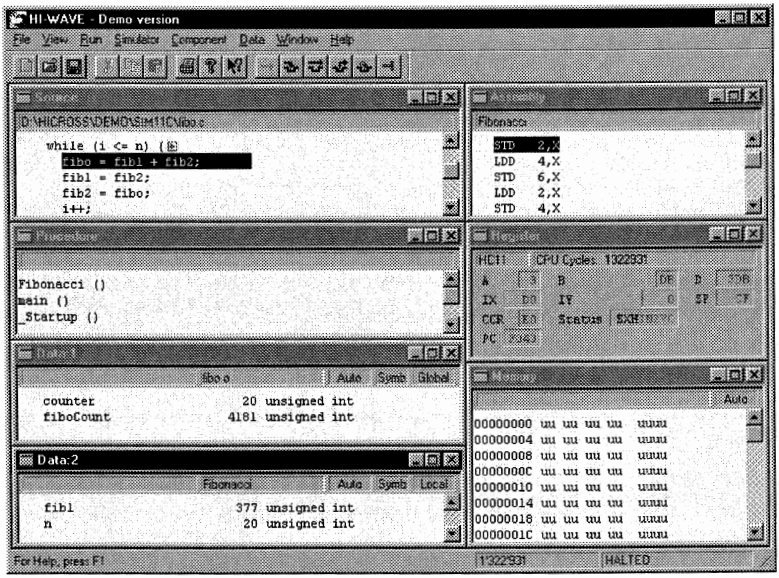
RUNNING Indicates in the status line that the application is running.

© *Copyright 1997 HIWARE*

Stopping an Application

1. Choose *Run* | *Halt* or click . The execution of the program is stopped.

The highlighted line The blue highlighted line in the *Source* window is the statement on which the program was stopped. (i.e. the next statement that will be executed.) The blue highlighted line in the *Assembly* window is the assembly instruction on which the program was stopped. (i.e. the next assembly instruction that will be executed.)



HALTED Indicates in the status line that the application was stopped.

Fibonacci This procedure is displayed on the first line of the *Procedure* window. It is the procedure currently being executed.

main This procedure is displayed on the second line of the *Procedure* window. It is the procedure which has invoked `procedure Fibonacci`.

Global In the heading line of the *Data:1* component indicates that global variables of module `FIBO` are displayed below.

© *Copyright 1997 HIWARE*

counter **20** Indicates that the current value of variable `counter` is 20.

Local In the heading line of the *Data:2* component indicates that local variables are displayed in this window.

Fibonacci In the heading line of the *Data:2* component indicates that local variables of `Fibonacci` are displayed in the window.

n **20** Indicates that the current value of variable `n` is 20.

Drag and Drop Facilities

To Drag and Drop an Object

1. Select the component containing the object you want to drag.
2. Make sure the destination component where you want to drag the object is visible.
3. Select the object you want.
4. Press and hold down the left mouse button, drag the object onto the destination component and then release the mouse button.

Drag and Drop Combinations

Drags and drops of objects are possible between different component windows. See below the most interesting possible combinations of drag and drop between components and associated actions. Other combinations are possible, see the *HI-WAVE User's Guide*.

Drag and Drop from Data to Memory

- Drag and drop `fiboCount` from the *Data:1* window to the *Memory* window.
- The memory where `fiboCount` is located is dumped. Its location is high-lighted in the *Memory* component.

Drag and Drop from Register to Memory

- Point to the `SP` register in the *Register* window.
- Drag and drop it in the *Memory* window. The memory starting at the address pointed by `SP` is dumped. The corresponding address is high-lighted in the *Mem-*

© *Copyright 1997 HIWARE*

ory component.

Drag and Drop from Procedure to Data:2 (Local)

- Select `main()` or `Fibonacci` in the *Procedure* window.
- Drag and drop it in the *Data:2* window. All the local variables of the dragged procedure are displayed in the *Data:2* window.

Drag and Drop from Procedure to Source

- Select `main()` or `Fibonacci` in the *Procedure* window.
- Drag and drop it in the *Source* window. The source corresponding to the dragged procedure is displayed.

View Where to Set Breakpoints

1. Click onto the title bar of the *Source* window. The *Source* window is activated and the *Source* menu appears in the menu bar.
2. Choose *Source* | *Marks*. All the statements where a breakpoint can be set are identified by a special mark: ◢

◢`fib2 = fibo;` Indicates in the *Source* window that a breakpoint can be set at this statement.

3. Choose *Source* | *Marks*. All marks are then removed.

Working with Breakpoints

Once the application is loaded, breakpoints may be set. There are four types of breakpoints:

temporary breakpoints The application will be stopped the next time it tries to execute the statement. This breakpoint is deleted as soon as it is reached.

permanent breakpoints The application will be stopped each time it tries to execute the statement. This breakpoint remains valid until the user explicitly deletes it.

counting breakpoints The application will be stopped once it has executed the statement a given number of times.

© *Copyright 1997 HIWARE*

conditional breakpoints The application will be stopped at the statement as soon as a given condition is true.

Setting Temporary Breakpoints

1. Point to statement `fib2 = fibo;` in the *Source* window,
2. Click right and choose *Run To Cursor* in the pull down menu. The program continues execution and stops before executing the statement.

BREAKPOINT In the status line indicates that the program was stopped due to a breakpoint.

3. Choose *Run* | *Start/Continue* or click .The program continues execution and will not stop any more.

Setting Permanent Breakpoints

1. Choose *Run* | *Halt* or click . The program is then stopped.
2. Point to the statement `return(fibo);` in the *Source* window.
3. Click right and choose *Set Breakpoint* in the pull down menu.

 This symbol in the *Source* window points to the statement in which the breakpoint is defined.

4. Choose *Run* | *Start/Continue* or click .The application continues execution and stops before executing the selected statement.

BREAKPOINT In the status line indicates that the program was stopped due to a breakpoint.

In our demo program, local variable `n` is incremented each time the program enters `procedure Fibonacci`. Please note the current value of this variable.

5. Choose *Run* | *Start/Continue* or click . The program continues execution and is stopped again the next time it comes to the breakpoint.

If you compare the current value of local variable `n` with the previous one, you will see that variable `n` was incremented by 1. `Procedure Fibonacci` was executed once again.

6. Point again to statement `return(fibo);` in the *Source* window.

© *Copyright 1997 HIWARE*

7. Click right and choose *Delete Breakpoint* in the pull down menu. The permanent breakpoint in the application is deleted.

Stepping

There are four kind of stepping actions:

single step The application will be stopped at the next source statement.

step over The application will be stopped at the next source statement, but function calls are skipped.

step out The application will be stopped at the next source statement in the caller function.

assembly step The application will be stopped at the next assembly statement.

Single Step on C Level

Choose *Run* | *Single Step* or click . The application stops at the next high-level language statement. This statement can be located in a function that is called.

Step Over

Click . The application stops at the next high-level language statement of the current function. Function invocations are skipped.

Single Step on Assembler Level

Choose *Run* | *Assembly Step* or click . The application stops at the next assembler statement.

Working with Data and Registers

The features of the *Data*, *Register* and *Memory* windows are described below.

Displaying and Updating Local Variables

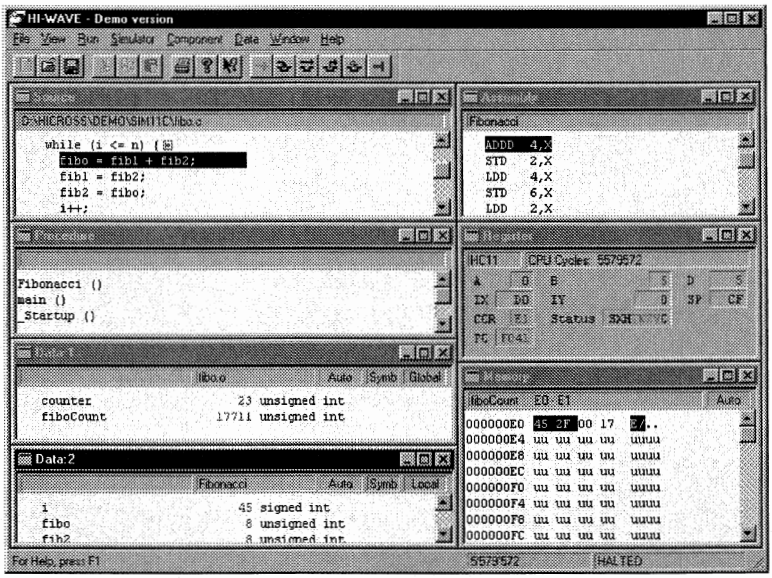
Local variables of procedures that are listed in the *Procedure* window can be dis-

played and eventually updated. (Note: this feature can only be used when the application is not running).

1. Double-click on `main` in the *Procedure* component.

Local In heading line of the *Data:2* component indicates that local variables are displayed.

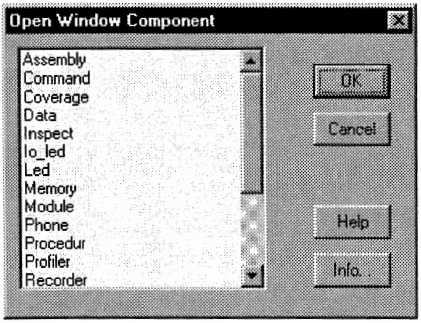
2. Double-click on variable `i` in the *Data:2* component.
3. Now you can change the value of this variable. Enter 45 and press return.
4. The value displayed next to variable `i` is now 45. This value has directly been updated in the target system's RAM if you are working on a target board.



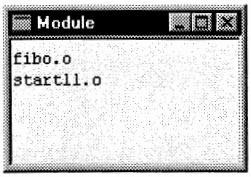
Displaying and Updating Global Variables

1. Choose *Component* | *Open...* . The *Open Window Component* dialog pops up.

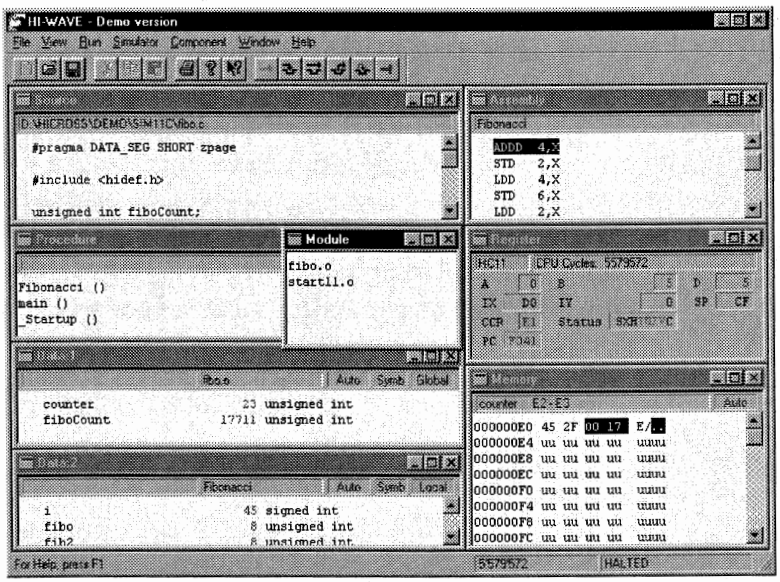
© *Copyright 1997 HIWARE*



2. Select *Module* and click *OK*. The *Module* window is now opened.



3. Global variables of modules that are listed in the *Module* window can be displayed and updated. In the *Module* window: double-click on `fibo.o`.



GLOBAL Indicates in the heading line of the *Data:1* component that global vari-

ables are displayed.

4. Double-click on variable `counter` in the *Data:1* window. You can change the value of this variable.
5. Press ESC: the value in variable `counter` remains unchanged.

Drag and Drop from Module to Data:1 Global

- Select `START11.O` in the *Module* component.
- Drag and drop it in the *Data:1* window. The global variables from the selected module are displayed in the *Data:1* window.

Drag and Drop from Module to Source

- Select `START11.O` in the *Module* component.
- Drag and drop it in the *Source* window. The source corresponding to the selected module is displayed in the *Source* window.

Displaying and Updating Register Contents

The *Register* window displays the current values of all the registers.

Point to register `A` in the *Register* window and double-click.

You can change the `A` register value (be careful, before updating any registers, you have to be sure that the new register contents do not disturb the application).

Press ESC to restore the previous value of the register.

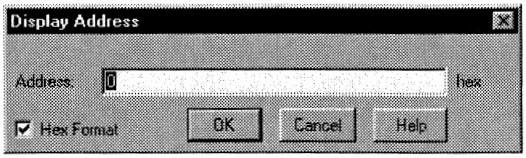
Inspecting Memory

In the *Memory* window, current content of the simulated target system's RAM starting at address `$00` is displayed in hexadecimal format.

The start address for global data defined in the Linker Parameter File `FIBO.PRM`. is address `$80`. The data should then be located starting at this location.

1. Click the title bar of the *Memory* window. The *Memory* window is activated and the *Memory* menu appears in the menu bar.
2. Choose *Memory* | *Address...* . The *Display Address* dialog is opened.

© *Copyright 1997 HIWARE*



3. Here you can change the base address for the memory display. Enter `80`. This location is reached in the *Memory* component.
4. Choose *Memory* | *Format* | *UDec*. Memory contents are shown in unsigned decimal format.
5. Other data formats are also available. Choose *Memory* | *Format* | *Hex* to reset the format to hexadecimal.

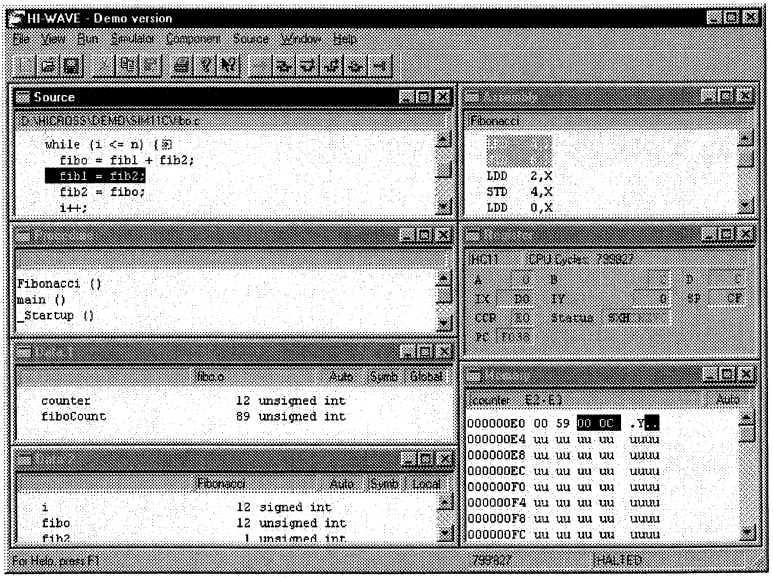
Working with the Assembler Window

On-line Disassembling

While the application is stopped you can display assembler statements associated with high-level language statements.

1. Double-click on `main()` in the *Procedure* component.
2. Point the mouse before the statement `fib1 = fib2;`
3. Click and hold down it while moving to the right.
4. When the statement is selected, unclick.
5. Point the mouse into the selection and drag it to the *Assembly* window. The drag and drop cursor appears on the screen.
6. When the *Assembly* window is reached, unclick. The assembler statements corresponding to the selected high-level statement are grayed.

© *Copyright 1997 HIWARE*



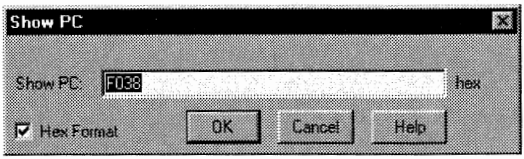
grayed lines The grayed lines in the *Assembly* window are assembler statements generated by the Compiler for the selected high-level language statement.

Assembler statements are disassembled directly from the simulated target system's ROM (or RAM). As a consequence, when, as an example, the code has been destroyed due to an address error, it is immediately visible.

Consulting the Code

1. Click on the title bar of the *Assembly* window. The *Assembly* window is activated and the *Assembly* menu appears in the menubar.
2. Choose *Assembly* | *Display Code*. Hexadecimal code associated with each assembler statement is displayed to the left of the statement.
3. Choose *Assembly* | *Display Adr*. The memory location in the simulated target system's ROM (or RAM) of each assembler statement is displayed.
4. Choose *Assembly* | *Display Code* from the menu again. The hexadecimal code is removed.
5. Choose *Assembly* | *Display Adr*. The memory locations are removed.
6. Choose *Assembly* | *Address...* . The *Show PC* dialog is opened.

© *Copyright 1997 HIWARE*



7. In this dialog you can define an address from where you want to have the code displayed.
8. Click *Cancel* to close this dialog.

Working with Breakpoints in the Assembly Window

Breakpoints can be set in the *Assembly* window in the same way as they are set in the *Source* window.

Simulating an Assembly Source File

The simulator offers the possibility to load an assembler application. The `MACRODEM.ASM` file must be assembled using the Macro Assembler and then linked (use `MACRODEM.PRM`). `MACRODEM.ABS` is then generated. This application copies some constant strings defined in ROM in some destination strings defined in RAM.

This demo can be run either with the debugger or the simulator.

To run it with the debugger, use the *HIWARE TOOLS* shell to switch to project directory `D:\HICROSS\DEMO\EVBU11A`.

To run it with the simulator, use the *HIWARE TOOLS* shell to switch to project directory `D:\HICROSS\DEMO\SIM11A`.

1. Start *HI-WAVE* using the *Debugger* button in the *HIWARE TOOLS* shell.
2. Choose *Simulator* | *Load* or *Monitor* | *Load* and load `MACRODEM.ABS` .
3. Click ⊞ in the *Data:1* component. All the variables of the application are displayed.
4. Click → to start the execution of the application. After a while, click ⊣ to stop.The sources strings have been copied to the destination strings.

The *Source* component displays the assembler source file. Break points can be set anywhere in this component or in the *Assembly* component. All functions available for C sources based applications are also available for assembler based applications.

© *Copyright 1997 HIWARE*

Simulator Specific Features

Previously described actions should be performed either using the debugger (*Smonitor* target) or the simulator (*Sim* target).

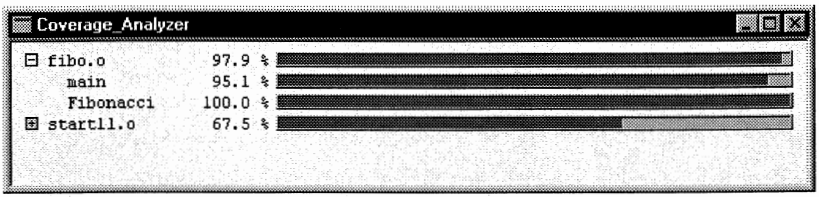
Following features are only available on the simulator (*Sim* target).

To run them use the *HIWARE TOOLS* shell to switch to the `D:\HICROSS\DEMO\SIM11C` project directory.

Coverage Functions

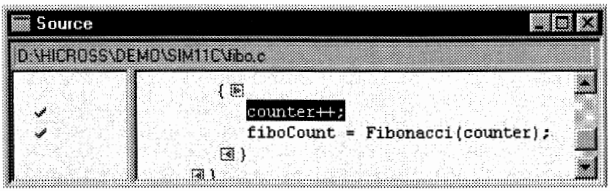
Coverage analysis can be introduced through the `FIBO.ABS` application:

1. Choose *File* | *Exit* to quit *HI-WAVE*.
2. Start *HI-WAVE* using the *Debugger* button in the *HIWARE TOOLS* shell.
3. Choose *Component* | *Set Target* and load the `Sim` (simulator) target.
4. Choose *Simulator* | *Load* and load `FIBO.ABS`. Click to start the execution of the application.
5. After a short while, click to stop the application.
6. Choose *Component* | *Open...* . The *Open Window Component* dialog pops up.
7. Select *Coverage* in the list box and click *OK*. The *Coverage Analyzer* component pops up. Coverage information about the different modules in the application are displayed in the component.



8. Click ⊞ . Coverage information about the different functions in the module are displayed.
9. Drag from the *Coverage* component to the *Source* or *Assembly* component. Coverage information about the source or assembly instructions are displayed in the destination component.

© *Copyright 1997 HIWARE*

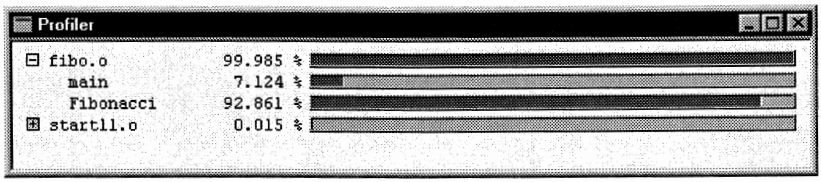


The ✓ symbol means that all statements on the source line it precedes has been executed.

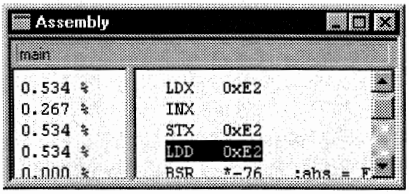
Profiling Functions

The *Profiler* component allows to determine how many time the application spend in each module, function or statement. Such information can be useful for optimization purpose.

1. Choose *Component* | *Open...* . The *Open Window Component* dialog pops up.
2. Select *Profiler* in the list box and click *OK*. Profiling information about the different modules in the application are displayed in the component.



3. Click ⊞ in front of a module name: profiling information about the different functions in the module are displayed in the component.
4. Drag from the *Profiler* component into the *Source* or *Assembly* component. Profiling information are displayed in the destination component.

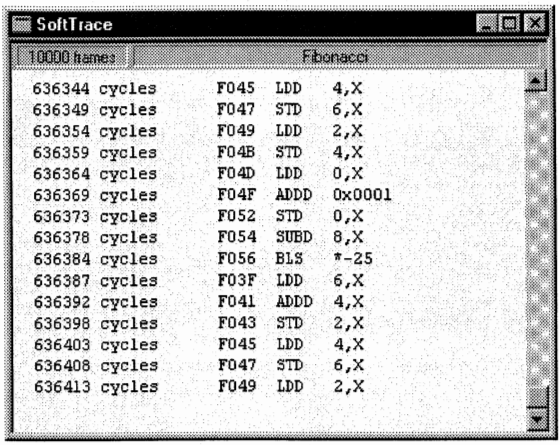


© *Copyright 1997 HIWARE*

Software Tracing Functions

The *SoftTrace* component records the last cycles executed in the application.

1. Choose *File* | *Exit* to quit *HI-WAVE*.
2. Start *HI-WAVE* using the *Debugger* button in the *HIWARE TOOLS* shell.
3. Choose *Component* | *Set Target* and load the `Sim` (simulator) target.
4. Choose *Simulator* | *Load* and load `FIBO.ABS`.
5. Choose *Component* | *Open...* . The *Open Window Component* dialog pops up.
6. Select *Softtrac* in the list box and click *OK*. The *SoftTrace* component window pops up.
7. Click to start the execution of the application.
8. After a short while, click to stop the execution.


SoftTrace
10000 frames
Fibonacci
636344 cycles F045 LDD 4,X
636349 cycles F047 STD 6,X
636354 cycles F049 LDD 2,X
636359 cycles F04B STD 4,X
636364 cycles F04D LDD 0,X
636369 cycles F04F ADDD 0x0001
636373 cycles F052 STD 0,X
636378 cycles F054 SUBD 8,X
636384 cycles F056 BLS *-25
636387 cycles F03F LDD 6,X
636392 cycles F041 ADDD 4,X
636398 cycles F043 STD 2,X
636403 cycles F045 LDD 4,X
636408 cycles F047 STD 6,X
636413 cycles F049 LDD 2,X

9. The recorded frames are displayed in the *SoftTrace* component.

Visualization Function

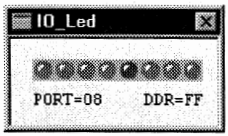
IO_LED Component

`LEDS.ABS` demo program switches the different leds in the *IO_LED* component. The leds are located at address `0x210`.

1. Choose *File* | *Exit* to quit *HI-WAVE*.
2. Start *HI-WAVE* using the *Debugger* button in the *HIWARE TOOLS* shell.
3. Choose *Component* | *Set Target* and load the `Sim` (simulator) target.

© *Copyright 1997 HIWARE*

4. Choose *Simulator* | *Load* and load `LEDS.ABS`.
5. Choose *Component* | *Open...* . The *Open Window Component* dialog pops up.
6. Select *IO_LED* in the list box and click *OK*. The *IO_LED* component window pops up.
7. Click to start the execution of the application. The leds are switched ON/OFF.

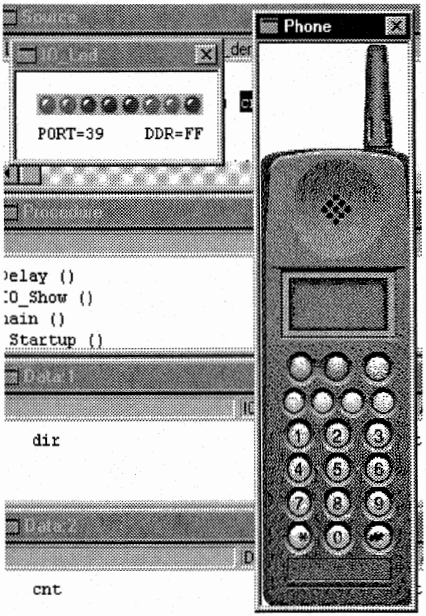


Telephone Component

`LEDS.ABS` demo program offers the possibility to switch ON/OFF the different leds in the *IO_LED* component by dialing on the *PHONE* component.

1. Choose *File* | *Exit* to quit *HI-WAVE*.
2. Start *HI-WAVE* using the *Debugger* button in the *HIWARE TOOLS* shell.
3. Choose *Component* | *Set Target* and load the `Sim` (simulator) target.
4. Choose *Simulator* | *Load* and load `LEDS.ABS`.
5. Choose *Component* | *Open...* . The *Open Window Component* dialog pops up.
6. Select *IO_LED* in the list box and click *OK*. The *IO_LED* component window pops up.
7. Using the same procedure, open the *Phone* component.
8. Click to start the execution of the application. After a while, click to stop.

© *Copyright 1997 HIWARE*



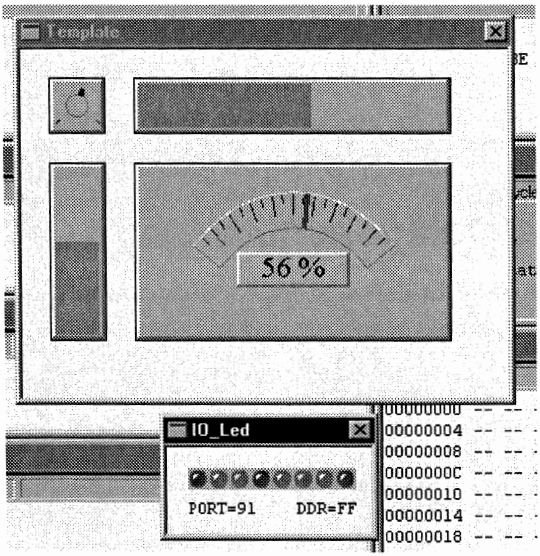
9. Click one of the button of the phone. The leds in *IO_LED* displays the ASCII code corresponding to the value of the phone button you have selected.

Template Component

`IO_DEMO.ABS` demo program switches the different leds in the *IO_LED* component and also changes the display in the *Template* Component.

1. Choose *File* | *Exit* to quit *HI-WAVE*.
2. Start *HI-WAVE* using the *Debugger* button in the *HIWARE TOOLS* shell.
3. Choose *Component* | *Set Target* and load the `Sim` (simulator) target.
4. Choose *Simulator* | *Load* and load `IO_DEMO.ABS` .
5. Choose *Component* | *Open...* . The *Open Window Component* dialog pops up.
6. Select *IO_LED* in the list box and click *OK*. The *IO_LED* component window pops up.
7. Using the same procedure, open the *Template* component.
8. Click to start the execution of the application.

© *Copyright 1997 HIWARE*



9. The leds are switched ON/OFF and the content of the *Template* component is periodically changed.

© Copyright 1997 HIWARE

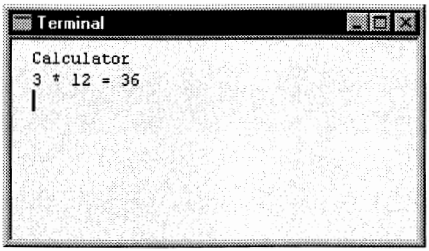
Debugging on the M68HC11EVBU2 Target Board

Following examples can only be run on the debugger (*Smonitor* target).

Using the *Terminal* Component

The `Calc` application offers the possibility to create a calculator with the *M68HC11EVBU2* target board. The application use the *Terminal* component to communicate with the user.

1. To run this demo, use the *HIWARE TOOLS* shell to switch to the `D:\HICROSS\DEMO\EVBU11C` project directory.
2. Connect the *M68HC11EVBU2* target board to your *PC* and start the debugger.
3. The *Download monitor* dialog is displayed. Press on the reset button on the target board. The monitor is loaded.
4. Choose *Monitor* | *Load* and load `Calcmoni.abs` .
5. Choose *Component* | *Open....* The *Open Window Component* dialog pops up. Select *Terminal* in the list box and click *OK*. The *Terminal* component is displayed.
6. Click to start the execution of the application.
7. `Calculator` is displayed in the Terminal component. Type in `3 * 12 =`. When typing = the result of the operation is displayed: `36`.



8. Click to stop.
9. The code is displayed in the *Source* component. You can use the *Module* component to select the `calc.o` source file. You can then insert breakpoint to debug the application.

© *Copyright 1997 HIWARE*

> *Note: For more information on how to switch between modules and how to debug an application, please refer to the section Debugger/Simulator of this demo guide.*

Debugging an Interrupt Driven Application

RTI Interrupt, C Based Application

The `RTIDEMO.C` example shows how to implement an interruption in C language Each time the RTI interrupt is issued, the `TimerInterrupt` procedure is entered. The `ticks` counter is then incremented.

Assignment between the RTI interrupt and the interrupt procedure is done in the Linker Parameter File `RTIDEMO.PRM` using the directive `VECTOR ADDRESS`:

```
VECTOR ADDRESS 0xFFF0 TimerInterrupt
```

The pragma `#pragma TRAP_PROC` in the C source file, defines the `TimerInterrupt` as trap (interrupt) function.

To run the RTI C language demo:

1. To run this demo, use the *HIWARE TOOLS* shell to switch to the `D:\HICROSS\DEMO\EVBU11C` project directory.
2. Connect the *M68HC11EVBU2* target board to your *PC* and start the debugger.
3. The *Download monitor* dialog is displayed. Press on the reset button on the target board. The monitor is loaded.
4. Choose *Monitor* | *Load* and load `RTIDEMO.ABS`.
5. Select the *Source* component window.
6. Choose *Source* | *Open Module*. Double-click on `RTIDEMO.C`. The `RTIDEMO.C` source file is displayed in the *Source* component.
7. Set a breakpoint into the `TimerInterrupt` function.
8. Click to start the execution of the application. The execution stops on the breakpoint.
9. Click again. Each time an RTI interrupt is generated, the `TimerInterrupt` procedure is entered and the value of `ticks` is incremented.

© *Copyright 1997 HIWARE*

RTI Interrupt, Assembler Application

The `RTIASM.ASM` example shows how to implement an interruption in assembler. Each time the RTI interrupt is issued, the `TimerInterrupt` procedure is entered. The `ticks` counter is then incremented.

Assignment between the RTI interrupt and the interrupt procedure is done in the Linker Parameter File `RTIASM.PRM` using the directive `VECTOR ADDRESS`:

```
VECTOR ADDRESS 0xFFF0 TimerInterrupt
```

The label `TimerInterrupt` is exported in the assembler source file using the directive `XDEF`.

To run the RTI assembler demo:

1. To run this demo, use the *HIWARE TOOLS* shell to switch to the `D:\HICROSS\DEMO\EVBU11A` project directory.
2. Connect the *M68HC11EVBU2* target board to your *PC* and start the debugger.
3. The *Download monitor* dialog is displayed. Press on the reset button on the target board. The monitor is loaded.
4. Choose *Monitor* | *Load* and load `RTIASM.ABS`.
5. Select the *Source* component window.
6. Set a breakpoint on an assembler statement after the `TimerInterrupt` label and before the `RTI` instruction.
7. Click ⊞ in the *Data:1* component. All the variables of the application are displayed.
8. Click to start the execution of the application. The execution stops on the breakpoint.
9. Click again. Each time an RTI interrupt is generated, the `TimerInterrupt` procedure is entered and the value of `ticks` is incremented.

© *Copyright 1997 HIWARE*

© Copyright 1997 HIWARE

© Copyright 1997 HIWARE

Ask for your local distributor:

HIWARE

Innovation for your Success

HIWARE · Riehenring 175 · 4058 Basel · Switzerland
Phone +41 61 690 75 00 · Fax +41 61 690 75 01
Email: info@hiware.hicom.ch
WWW: http://www.hiware.com